



Ecole Nationale Supérieure des Télécommunications
Département Traitement du Signal et des Images
Ecole doctorale Edite

eman ta zabal zazu



Universidad del País Vasco Euskal Herriko Unibertsitatea
University of the Basque Country
Computer Engineering Faculty
Intelligent Systems Group

THÈSE DE DOCTORAT
SPÉCIALITÉ SIGNAL ET IMAGES

Inexact Graph Matching Using
Estimation of Distribution Algorithms

Mise en Correspondence Inexacte de Graphes par
Algorithmes d'Estimation de Distributions

Endika Bengoetxea

Directeurs de Thèse : Isabelle Bloch
Pedro Larrañaga

19 décembre 2002

Acknowledgements

There are some people to whom I am deeply indebted for their help in the process that ends with this PhD dissertation. I take this opportunity to acknowledge their never faltering encouragement and support.

First of all, I wish to thank my two PhD supervisors Isabelle Bloch and Pedro Larrañaga for offering me the opportunity to do this PhD in collaboration between their laboratories. Special thanks also for their wise supervision, guidance, patience, and friendly encouragement.

I also owe a debt of gratitude to my colleagues of the Intelligent Systems Group of the Department of Computer Science and Artificial Intelligence of the University of the Basque Country in San Sebastian, as well as to the members of the Image Processing and Interpretation Group from the Department of Signal and Image Processing at the Ecole Nationale Supérieure des Télécommunications for their help on the development of this thesis, their friendly welcome, and their patience with my spoken and written French.

My gratitude also to Roberto Cesar, from the University of São Paulo in Brazil, for his helpful collaboration and dedication in many of the image processing parts of this thesis as well as in providing me the data required for a successful completion of the experiments. I would like to thank Jose Miguel and Alex Mendiburu from the Department of Computer Architecture and Technology for their support and contribution in the parallel programming section, as well as Aymeric Perchant, Claudia Boeres, and Jérémie Pescatore for their useful advice and contribution to this thesis.

Special thanks also for the members of the Deanship of the Computer Engineering Faculty at the University of the Basque Country, specially to the Dean Iñaki Morlan and the Vice-dean Teresa Miquélez, for their understanding and confidence on me while I was working in the Deanship and on this PhD thesis at the same time. They, as well as many other colleagues, family and friends have *suffered* the stress of carrying out this PhD thesis in three years and at the same time being working as a lecturer, Erasmus coordinator, and Academic Secretary in my home Faculty.

Also, I would like to thank a lot the interest and the useful comments of the two reviewers of the PhD dissertation Jean-Michel Jolion and Michele Sebag, as well as the rest of the members of the jury Olivier Cappe, Yves Sorel and Jose Antonio Lozano for the time dedicated for the evaluation of this PhD.

Finally, the author would like to acknowledge the financial help and the support of these institutions which have supported partially this dissertation:

- The Spanish Ministry for Science and Education, with project HF1999-0107
- The French Ministry for Education, Research and Technology with project Picasso-00773TE.
- The Spanish Ministry for Science and Technology with project TIC2001-2973-C05-03.
- The University of the Basque Country, with the projects UPV140.226-EB131/99 and 9/UPV/EHU 00140.226-12084/2000
- The Department of Education Universities and Research of the Basque Government, with project PI 1999-40

Summary in Spanish – Resumen en castellano

Motivación

Desde hace ya varias décadas se han utilizado las computadoras para trabajar con imágenes. Los ejemplos más clásicos son aquellos en los que la computadora mejora una imagen con ruido o que está borrosa. Sin embargo, este tipo de trabajos utiliza información que contiene la imagen que se podría considerar de *bajo nivel*, es decir información tal que nivel de gris de los pixels, segmentación, e incluso formas geométricas de segmentos de una imagen. Sin embargo, también encontramos en las imágenes otro tipo de información que podríamos denominar de alto nivel, como son por ejemplo la estructura de la escena en la imagen, la topología, y organización espacial de los diferentes objetos en la imagen. Este último tipo de información la que resulta más difícil de procesar para una computadora, y es precisamente parte del objetivo de esta tesis: presentar un procedimiento novedoso para facilitar el reconocimiento automático de imágenes.

En los últimos años se ha propuesto la utilización de grafos para representar el conocimiento estructural utilizando técnicas matemáticas, y poder así utilizar esta representación para facilitar la comprensión a las computadoras. El interés de este tipo de representaciones va en aumento entre la comunidad científica debido a la posibilidad de utilizar esta representación junto con algoritmos de correspondencia de grafos¹, ya que esta representación es muy versátil y permite también representar las variaciones y diferencias estructurales entre diferentes objetos.

Asimismo se han propuesto diversas aplicaciones directas de este tipo de representación para realizar el reconocimiento automático de imágenes. En este tipo de aplicaciones, el conocimiento sobre posibles variaciones estructurales del objeto que aparece en las imágenes (una persona, un rostro, una imagen médica, etc.) es expresado por medio de un modelo que se representa como un grafo (o atlas, como también se denomina en ciertos casos). Ejemplos de áreas de reconocimiento de imágenes en las cuales este tipo de representaciones han sido utilizadas y publicadas son cartografía, reconocimiento de caracteres, y reconocimiento de estructuras cerebrales a partir de imágenes de resonancia magnética. Este último ejemplo se ilustra en la Figura 1.

El *grafo modelo* se construye habitualmente utilizando un vértice para expresar cada una de las regiones del objeto a reconocer (Ej.: en el caso de imágenes de un cerebro humano, habrá un vértice para representar el cerebelo, otro para el cuerpo caloso...) y aristas para representar la interrelación entre estas regiones. Asimismo, se utilizan atributos para expresar las propiedades de vértices y aristas para poder así identificarlos y distinguir los unos de los otros. Este grafo modelo contiene atributos en los vértices y aristas, y muchas veces es necesaria la supervisión de un experto (Ej. un neurólogo para construir el modelo que representa un cerebro humano) para asegurar la veracidad e idoneidad del modelo construido.

Tras la construcción de un grafo modelo, y para poder realizar el reconocimiento de imágenes a través del mismo, es necesario construir un grafo a partir de cada una de las imágenes a reconocer. Estos grafos que se denominan en la literatura *grafos de datos* o *grafos de entrada*, son construidos a menudo automáticamente por la computadora sin asistencia del usuario. En este proceso de construcción del grafo de datos correspondiente a una imagen, uno de los pasos más significativos en cuanto al rendimiento del método es el de la segmentación de la propia imagen previo a la construcción del propio grafo: el grafo de datos se genera utilizando un vértice para representar cada uno de los segmentos en los que

¹La correspondencia de grafos se denomina en inglés *graph matching*

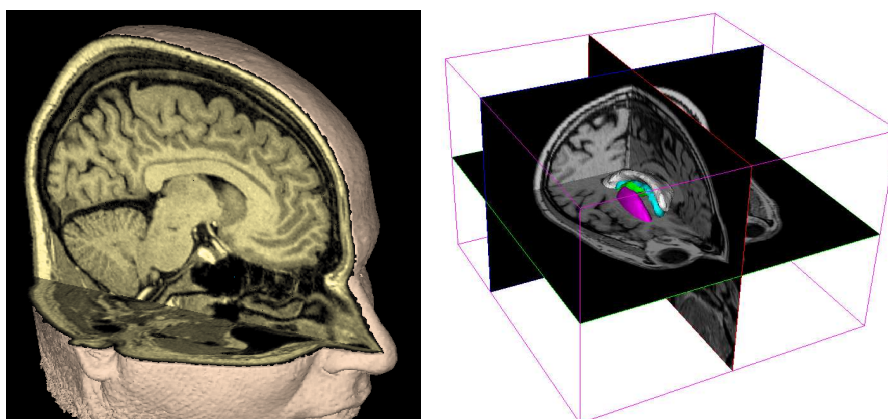


Figure 1: Ejemplo de aplicación de la correspondencia entre grafos al reconocimiento de imágenes. El objetivo en este problema en concreto es el de identificar las diferentes partes del cerebro a partir de la imagen en 3D de la izquierda.

se ha dividido la imagen (de forma similar al grafo modelo), y es precisamente debido a esto que la segmentación debe hacerse con especial atención para asegurar un número y tamaño adecuado de los segmentos.

Correspondencia de grafos

El reconocimiento de la imagen se realiza mediante la correspondencia entre los grafos modelo y de datos². Este proceso es equivalente a asignar una etiqueta a cada una de las regiones de la imagen a reconocer, de manera que se designa cada región de la imagen como perteneciente a una sección concreta del objeto. Esta asignación de etiquetas se efectúa vértice a vértice de entre los del grafo de datos, asignando a cada uno un vértice del grafo modelo. Este proceso se ilustra en la Figura 2.

Formalizando lo anterior, se definen dos grafos en problemas de reconocimiento de patrones basados en modelos, –el grafo modelo G_M y el grafo de datos G_D – y el procedimiento de buscar correspondencias entre ellos se basa en analizar las semejanzas entre ellos según sus vértices, aristas y atributos. Por lo tanto, podemos plantear el problema de correspondencia entre grafos de la siguiente manera: dados dos grafos $G_M = (V_M, E_M)$ y $G_D = (V_D, E_D)$, el problema consiste en encontrar un homomorfismo $h : V_D \rightarrow V_M$ tal que $(u, v) \in E_D$ si y sólo si $(f(u), f(v)) \in E_M$.

Se han planteado varios problemas en la literatura para poder efectuar la correspondencia entre grafos y demostrar la validez de las distintas técnicas. En los problemas más simples se asume que el número de vértices en ambos grafos es el mismo, por lo que la técnica empleada en buscar la mejor correspondencia o solución debe devolver un isomorfismo. Este tipo de problemas se engloban dentro de *correspondencias exactas de grafos*, y la propiedad que se cumple en estos es $|V_M| = |V_D|$. Sin embargo, en muchos otros problemas (sobre todo en aquellos en los que la segmentación se ha realizado automáticamente) la condición de isomorfismo es demasiado estricta para poder satisfacerla y el número de vértices del grafo de datos es habitualmente superior al del modelo. En este otro tipo de problemas, cada posible solución es un homomorfismo h en el que se asigna a cada vértice del grafo de datos

²A este proceso de correspondencia entre grafos también se le denomina en castellano *macheo de grafos*.

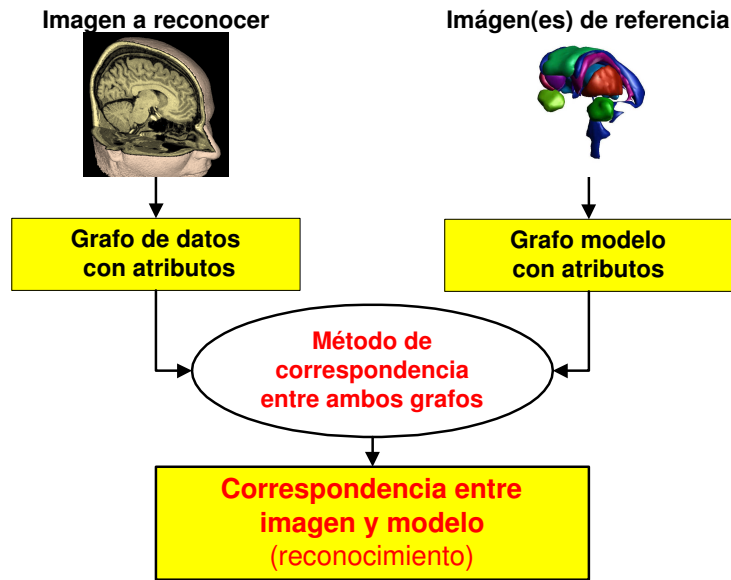


Figure 2: Ejemplo de cómo se realiza el reconocimiento de imágenes a partir de los grafos modelo y de datos.

una etiqueta correspondiente a un vértice. Estos últimos son conocidos como problemas de *correspondencia inexacta de grafos*, y su complejidad es mucho mayor que la de los anteriores. En este último caso se cumple la propiedad $|V_M| < |V_D|$.

Existen también otras posibles formas de clasificar los problemas de correspondencia de grafos. En cualquier caso, la clasificación basada en problemas de correspondencia de grafos exactos e inexactos es la más extendida. La Figura 3 muestra la clasificación más difundida en la literatura.

Se han propuesto muchos tipos diferentes de problemas de correspondencia de grafos en la literatura, y en muchos de ellos se utilizan los siguientes enfoques:

- **Nodo nulo o *dummy*:** son problemas en los que se utiliza un nodo adicional para tener una correspondencia añadida y poder dejar de considerar regiones en la imagen que no correspondan a partes del objeto a reconocer.
- **Correspondencias múltiples entre nodos:** este tipo de problemas son mucho más complejos de los comentados hasta ahora ya que es posible que un segmento de la imagen a reconocer sea a la vez parte de más de una región del objeto a analizar. Aunque este tipo de problemas pueden plantearse, su resolución conlleva más combinaciones y el espacio de búsqueda para encontrar la solución óptima puede llegar a ser intratable computacionalmente.

Se pueden aplicar muchas técnicas diferentes para encontrar la correspondencia óptima en un problema concreto, incluso procedentes de paradigmas muy diversos. Ejemplos de artículos mostrando métodos aplicados a correspondencia entre grafos son los que proponen árboles de decisión [Messmer and Bunke, 1999], redes neuronales [Riviere et al., 2002], algoritmo EM [Cross and Hancock, 1998], relajación probabilística [Christmas et al., 1995], heurísticos y metaheurísticos [Pelillo et al., 1999], algoritmos genéticos [Wilson and Hancock, 1997], y programación evolutiva [Singh and Chaudhury, 1997]. Muchas de estas técnicas están

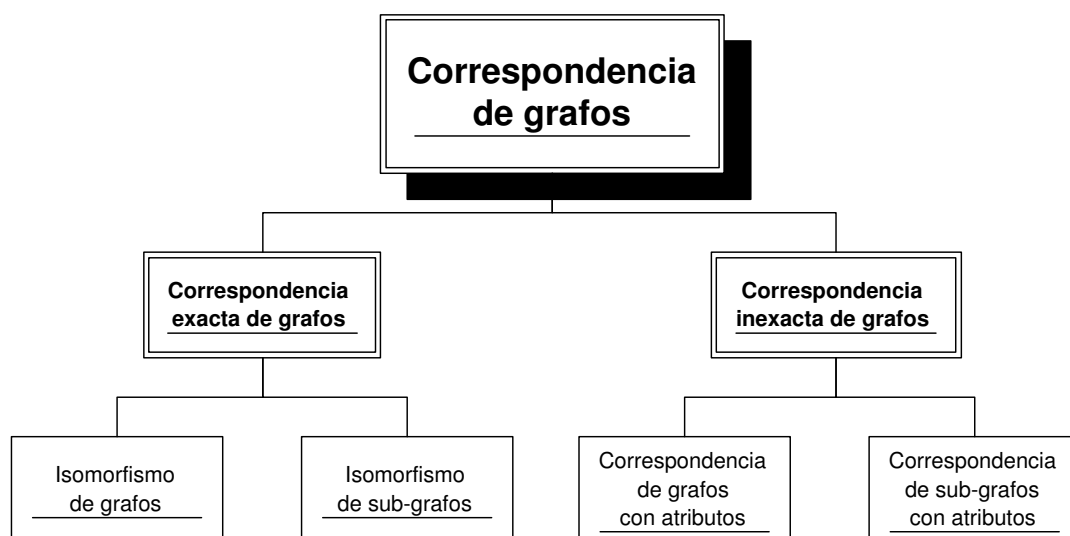


Figure 3: Clasificación de problemas de correspondencias entre grafos en dos clases principales.

orientadas a correspondencias exactas de grafos y por lo tanto devuelven el isomorfismo que representa la mejor correspondencia entre el grafo modelo y el de datos generado a partir de la imagen original a ser reconocida. Sin embargo, en muchos otros (sobre todo en aquellos aplicados a problemas con imágenes reales) la condición de isomorfismo es demasiado fuerte y por lo tanto se pueden encontrar en la literatura numerosas aproximaciones para resolver problemas de correspondencias inexactas de grafos.

Los problemas que se han considerado en esta tesis doctoral son todos problemas de correspondencia inexacta de grafos, y en ningún caso se ha considerado la posibilidad de añadir nodos nulos. Además, con el objetivo de no aumentar la complejidad de problemas que ya de por sí son complejos, se ha decidido realizar una sobre-segmentación de la imagen a reconocer para evitar la posibilidad de correspondencias múltiples entre grafos. De esta forma el número de vértices de los grafos de nodos es mayor en nuestros problemas. El método que proponemos para realizar la correspondencia entre grafos es el de Algoritmos de Estimación de Distribuciones (EDAs), tema que se estudia en profundidad en esta tesis. Así, se puede resumir el objetivo y la motivación de esta tesis como la presentación de un nuevo paradigma para hacer frente a correspondencias inexactas de grafos aplicados al dominio de la visión por computador y reconocimiento de patrones.

Correspondencia de grafos planteado como un problema de optimización combinatorial

El problema de correspondencia inexacta de grafos es NP-duro, tal y como demuestran varios trabajos en la literatura. Debido a esta complejidad tan grande, el uso de algoritmos que proporcionan una aproximación a la mejor solución se muestra necesario. Así, con el objetivo de poder aplicar este tipo de algoritmos, se puede plantear el problema de correspondencias entre grafos como un problema de optimización combinatorial con restricciones. Esta tesis analiza los aspectos y mecanismos que deben aplicarse para poder plantear el problema de esta manera y poder aplicar posteriormente algoritmos como los denominados Algoritmos de Estimación de Distribuciones (EDAs).

Para poder plantear un problema cualquiera como un utilizando técnicas este tipo de algoritmos se necesita esencialmente definir las siguientes características:

- **Una representación de los individuos** o soluciones, de manera que podamos expresar cada uno de los puntos en el espacio de búsqueda.
- **Una función objetivo a optimizar**, que asigna un valor a cada solución posible (o individuo) para expresar cómo es de adecuado el individuo analizado como solución para el problema.

Los individuos se expresan como vectores de valores que pueden ser discretos o continuos. Algunos algoritmos están diseñados para trabajar únicamente con individuos discretos o continuos, aunque otros como por ejemplo los EDAs permiten utilizar ambos tipos de individuos. Esta tesis presenta la forma de trabajar con ambos tipos de individuos y diferentes representaciones dentro de los dominios discretos y continuos para aplicarlas en problemas de correspondencia de grafos.

Un ejemplo de una de las representaciones que se ha aplicado en varios problemas expuestos en esta tesis consiste en asociar a cada vértice de G_D un vértice de G_M . Por lo tanto, el tamaño de los individuos en este caso concreto es de $n = |V_D|$ variables³, $X_i \in \mathbf{X}$ $i = 1, \dots, |V_D|$, donde cada variable contiene un valor entre 1 y $|V_M|$. El valor de cada variable en el individuo tiene el siguiente significado: $X_i = k$ $1 \leq i \leq |V_D|$, $1 \leq k \leq |V_M| \Leftrightarrow$ el i -ésimo vértice de G_D es identificado como el k -ésimo vértice de G_M .

Esta tesis propone otras dos representaciones más en el dominio discreto, así como una representación de individuos para el dominio continuo. Además, y con la intención de mostrar la robustez del método propuesto, se ha añadido una restricción a los problemas que deben de cumplir los individuos para poder tenerse en consideración. Esta restricción es la siguiente:

$$\forall a_M \in V_M \exists a_D \in V_D \mid h(a_D) = a_M$$

De esta forma, se espera que la solución devuelta por el método haya encontrado cada una de las partes representadas en el modelo al menos a uno de los segmentos de la imagen a reconocer. Esta restricción aumenta así la complejidad del problema, y esta tesis presenta varios mecanismos que pueden utilizarse para tener en cuenta restricciones mediante la aplicación de EDAs.

Finalmente, y con el objetivo de evaluar la bondad del individuo como posible solución a un problema, se definen dos funciones $c_N(a_D, a_M)$ y $c_E(e_D, e_M)$ que miden respectivamente la semejanza entre dos vértices $a_D \in V_D$ y $a_M \in V_M$, y entre dos aristas $e_D \in E_D$ y $e_M \in E_M$. Se definen en esta tesis un total de cinco funciones objetivo basadas en estas dos medidas de semejanza para poder medir la conveniencia de la solución que representa cada individuo para un problema determinado.

Algoritmos de Estimación de Distribuciones

Los Algoritmos de Estimación de Distribuciones (o EDAs como se les conoce en la literatura) es un tópico reciente dentro de la familia de la computación evolutiva aplicada a problemas de optimización. Otros ejemplos dentro del dominio de la computación evolutiva son los Algoritmos Genéticos (GAs).

³Cuando se aplican algoritmos genéticos se dice que los individuos están formados por *genes*, mientras que en otros paradigmas como en el caso de los EDAs se dice que tienen *variables*. En esta tesis se considera que los individuos constan de variables.

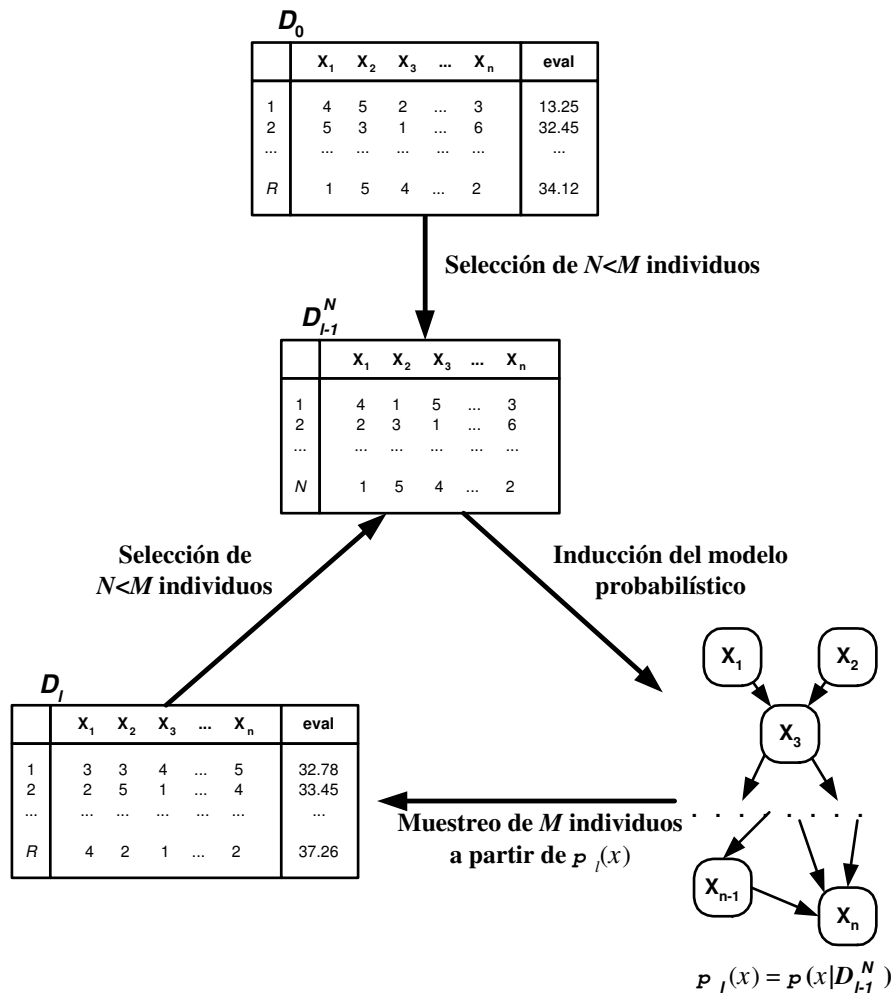


Figure 4: Ilustración de la aproximación EDA en procesos de optimización.

La computación evolutiva se engloba dentro del tipo de técnicas no deterministas, y los algoritmos de computación evolutiva se caracterizan por mantener un conjunto de posibles soluciones que se hace evolucionar progresivamente. Debido al símil con poblaciones de seres vivos, se denomina habitualmente *individuo* a cada una de las soluciones, *población* al conjunto de individuos, y *generación* a cada una de las poblaciones que evolucionan sucesivamente una tras otra. La diferencia más significativa entre los GAs (que son sin duda los más conocidos y utilizados dentro de la computación evolutiva) y los EDAs es que la evolución desde una generación a la siguiente se realiza en el caso de los GAs mediante operaciones de mutación y cruce, mientras que en los EDAs esta evolución se realiza utilizando técnicas basadas en la teoría de la probabilidad y más concretamente mediante el aprendizaje y la simulación de redes Bayesianas o redes Gaussianas. Esta idea se ilustra en la Figura 4.

La Figura 5 muestra el pseudocódigo genérico de los EDAs, que sigue esencialmente los siguientes pasos:

1. Primeramente, se genera la población inicial D_0 formada por R individuos. La creación de estos R individuos se realiza a menudo asumiendo una distribución uniforme en cada variable. Tras generar los individuos, estos se evalúan mediante la aplicación de

EDA

$D_0 \leftarrow$ Generar R individuos (la población inicial D_0) al azar

Repetir para $l = 1, 2, \dots$ hasta satisfacer un criterio de parada

$D_{l-1}^N \leftarrow$ Seleccionar $N < R$ individuos de D_{l-1} siguiendo un método de selección determinado

$\rho_l(\mathbf{x}) = \rho(\mathbf{x}|D_{l-1}^N) \leftarrow$ Estimar la distribución de probabilidad de que un individuo se encuentre entre los individuos seleccionados

$D_l \leftarrow$ Muestrear R individuos (la nueva población) a partir de $\rho_l(\mathbf{x})$

Figure 5: Pseudocódigo genérico de los EDA.

la función objetivo.

2. Segundo, para evolucionar la $l - 1$ -ésima población D_{l-1} hacia la siguiente D_l , se seleccionan N individuos ($N < R$) de D_{l-1} siguiendo un criterio. Denominamos D_{l-1}^N al conjunto de los N individuos seleccionados de la generación número $l - 1$.
3. Tercero, se induce el modelo gráfico probabilístico n -dimensional que mejor representa las interdependencias entre las n variables. Este paso es conocido como el del *aprendizaje*, y es el más crucial de los EDA debido a la importancia de tener en cuenta todas las dependencias entre variables para asegurar una evolución satisfactoria hacia individuos más válidos.
4. Finalmente, la nueva población D_l se constituye con R nuevos individuos obtenidos tras *simular* la distribución de probabilidad aprendida en el paso previo. A menudo se utiliza una aproximación elitista, manteniendo así el mejor individuo de la población D_{l-1}^N e la nueva población D_l . En este último supuesto, se crean cada generación un total de $R - 1$ nuevos individuos en vez de R .

Los pasos 2, 3 y 4 se repiten hasta satisfacer una condición de parada concreto. Ejemplos de criterios de parada son: llegar a un número de generación máxima, alcanzar un número máximo de individuos analizados, uniformidad en la población recién generada, o el hecho de no obtener un individuo con un valor de función objetivo mejor tras un cierto número de generaciones.

Se han propuesto una gran variedad de algoritmos en la literatura que son parte de los EDA, los cuales pueden clasificarse en tres grandes grupos dependiendo de la complejidad del tipo de dependencias entre variables que tienen en cuenta:

- **Sin interdependencias entre variables:** estos EDAs se basan únicamente en distribuciones univariantes $\rho(x_i)$. Esto significa que la estructura en forma de red Bayesiana (o Gaussiana si trabajamos en el dominio continuo) es fija y no contiene arcos. En otras palabras, esto significa que todas las variables del individuo se consideran independientes entre sí.

Como ejemplo de algoritmos pertenecientes a este grupo tenemos en el dominio discreto el llamado UMDA (Univariate Marginal Distribution Algorithm) [Mühlenbein, 1998]

donde la estimación de la distribución de probabilidad se realiza de la siguiente manera:

$$p(\mathbf{x}) = \prod_{i=1}^n p(x_i)$$

Otro ejemplo es el conocido como UMDA_c (Univariate Marginal Distribution Algorithm - continuous) [Larrañaga et al., 2000], el cual es el equivalente a UMDA aunque en este caso corresponde al dominio continuo.

- **Dependencias a pares entre variables:** los EDAs pertenecientes a este grupo están basados en distribuciones univariantes $p(x_i)$ y también en condicionales de segundo orden $p(x_j | x_i)$. La diferencia más significativa con respecto al grupo anterior es que la estructura de la red Bayesiana (o Gaussiana) puede ser diferente, aunque cada una de las variables puede tener como mucho un padre. Esto requiere un paso previo de selección de la mejor estructura que no existía en los anteriores.

Un ejemplo de EDAs discretos pertenecientes a este grupo es MIMIC (Mutual Information Maximization for Input Clustering) [de Bonet et al., 1997], que propone realizar la siguiente factorización de la probabilidad:

$$p(\mathbf{x}) = p(x_{i_1} | x_{i_2}) \cdot p(x_{i_2} | x_{i_3}) \cdots p(x_{i_{n-1}} | x_{i_n}) \cdot p(x_{i_n})$$

MIMIC se basa en buscar la permutación $\pi = (i_1, i_2, \dots, i_n)$ que minimiza la divergencia de Kullback-Leibler entre la estimación $\hat{p}_\pi(\mathbf{x})$ y la distribución real $p(\mathbf{x})$.

De nuevo, existe una versión continua de MIMIC llamada MIMIC_c (Mutual Information Maximization for Input Clustering - continuous) [Larrañaga et al., 2000].

- **Dependencias múltiples entre variables:** los EDAs pertenecientes a este grupo consideran tanto distribuciones univariantes como condicionales orden dos o superior, y por lo tanto las estructuras de redes Bayesianas o Gaussianas no tiene ninguna restricción en el número de arcos que pueden contener. Esta característica requiere una búsqueda exhaustiva de la mejor estructura gráfica probabilística entre todas las posibles, y por lo tanto estos algoritmos son más costosos en tiempo de ejecución que los de los grupos anteriores, aunque también son capaces de aprender modelos que reflejan más fielmente las interrelaciones entre las diferentes variables que forman parte de los individuos.

En el dominio discreto, como un ejemplo de EDAs pertenecientes a este grupo tenemos el conocido como EBNA (Estimation of Bayesian Networks Algorithm) [Etzeberria and Larrañaga, 1999]. Un ejemplo del dominio continuo es EGNA (Estimation of Gaussian Networks Algorithm) [Larrañaga et al., 2000, Larrañaga and Lozano, 2001], que sigue una aproximación similar a EBNA.

En EBNA se define un *score* o medida basada en la máxima verosimilitud penalizada conocida como BIC (Bayesian Information Criterion) [Schwarz, 1978] que mide la idoneidad de una estructura para representar las interdependencias entre los individuos. Utilizando esta medida, se busca la red Bayesiana que lo maximiza, y para ello los autores proponen el método conocido como Algoritmo B [Buntine, 1991]. Una vez definida la estructura, la factorización de la probabilidad se realiza de la siguiente forma:

$$p(\mathbf{x}) = \prod_{i=1}^n p(x_i | \mathbf{pa}(x_i))$$

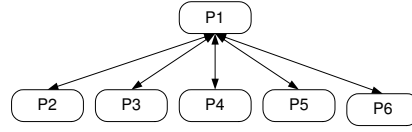


Figure 6: Esquema de ejecución maestro-esclavo, donde un proceso hace de *gestor* de tareas y los demás realizan partes de un trabajo común.

donde $\mathbf{pa}(x_i)$ es el conjunto de padres de la variable x_i en la red Bayesiana.

Son varios los algoritmos y aproximaciones de EDAs que se han propuesto en la literatura, aunque por el momento no hay demasiados artículos demostrando todo su potencial en comparación con otros paradigmas más conocidos como por ejemplo los GAs y Estrategias evolutivas (ES). Esta tesis viene a cubrir este espacio para la aplicación concreta de reconocimiento de objetos en imágenes. Otro de los aspectos novedosos de esta tesis es precisamente el hecho de aplicar EDAs por primera vez a problemas de correspondencia de grafos, y esta tarea se realiza para dominios discretos y continuos. Asimismo se han desarrollado en esta tesis métodos orientados a los EDAs para gestionar las restricciones que puedan existir en problemas, aplicables incluso en aquellos problemas que no sean de correspondencia de grafos.

En el caso de aplicar EDAs a correspondencia de grafos, las distribuciones y estructuras probabilísticas que se estiman en los EDA representan las dependencias entre las diferentes posibilidades de correspondencias entre vértices de G_D con respecto a vértices de G_M .

Uno de los mayores inconvenientes que se han encontrado al aplicar EDAs a correspondencias entre grafos es el hecho de que, debido al gran tamaño de los vértices y de los atributos a tener en cuenta en problemas reales, se requiere mucho tiempo de cálculo para que los EDAs evolucionen. Este inconveniente es especialmente evidente en los EDAs del tercer grupo dado que para la búsqueda de la mejor estructura gráfica probabilística se requiere analizar muchas posibles estructuras. Debido a esto, esta tesis propone técnicas de paralelismo para EDAs orientadas a reducir estos tiempos de ejecución.

Paralelización de EDAs

Esta tesis presenta un estudio que analiza las necesidades de CPU de los diferentes EDAs. Este análisis se ha realizado con la herramienta GNU *gprog*. En estos estudios se evidencia que los algoritmos más costosos computacionalmente son los del tercer grupo, y es precisamente el paso del aprendizaje el que mayor peso conlleva. En el caso concreto del problema de reconocimiento de estructuras cerebrales, el aprendizaje con EBNA supone el 85,7% del tiempo de ejecución total. Se decidió paralelizar el algoritmo EBNA debido a que los dominios discretos están más extendidos entre los EDAs y que los EBNA pertenecen al tercer grupo de los EDAs.

El aprendizaje en EBNA se basa en la medida BIC, y es precisamente esta medida la que requiere casi todo el tiempo de cálculo del aprendizaje. Una importante propiedad de BIC $-BIC(S, D)$ donde S es la estructura y D los datos a partir de los cuales se ha realizado el aprendizaje— es que se puede descomponer en componentes $BIC(i, S, D)$ que expresan la

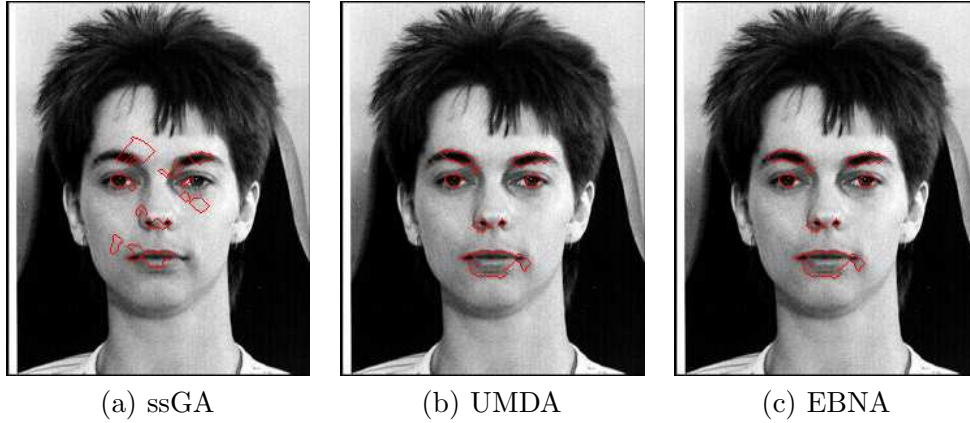


Figure 7: Ejemplo de un problema real de reconocimiento de estructuras faciales utilizando correspondencia entre grafos. Se ilustran los resultados conseguidos utilizando algoritmos genéticos y los EDAs UMDA y EBNA. Estos resultados muestran que el reconocimiento en este caso es superior en el caso de ambos EDAs.

medida local BIC para la variable X_i :

$$BIC(S, D) = \sum_{i=1}^n BIC(i, S, D) = \sum_{i=1}^n \sum_{j=1}^{q_i} \sum_{k=1}^{r_i} N_{ijk} \log \frac{N_{ijk}}{N_{ij}} - \frac{1}{2}(r_i - 1)q_i$$

Para paralelizar el programa se utilizó el modelo de ejecución maestro-esclavo ilustrado en la Figura 6, de manera que cada uno de los esclavos computan los términos $BIC(i, S, D)$ correspondientes a las diferentes variables X_i $i = 1, \dots, n$.

Para probar distintas técnicas de paralelismo se utilizaron las librerías pthreads y MPI, las cuales permitieron comparar la posibilidad de utilizar memoria compartida y paso de mensajes respectivamente con diferentes configuraciones de máquinas.

Los experimentos realizados mostraron que tanto con threads como con MPI se obtienen reducciones sustanciales en el tiempo de cálculo, llegando en algunas ocasiones a reducir el tiempo de cálculo inicial en un 60%.

Ejemplos experimentales

Esta tesis muestra diferentes ejemplos de aplicación de EDAs a problemas de correspondencia inexacta de grafos. De entre los ejemplos se presentan tanto problemas creados artificialmente para mostrar varias características de los EDAs y técnicas propuestas, así como problemas reales. Los estudios que se han realizado con ejemplos artificiales son los siguientes:

- Comparativa entre EDAs, GAs y ES para problemas de complejidad muy diferente.
- Análisis de cuatro métodos adaptados a los EDAs para controlar el cumplimiento de restricciones en problemas de correspondencia de grafos.
- Estudio de la evolución de las estructuras gráficas probabilísticas de generación en generación durante una búsqueda con EDAs discretos o continuos.
- Aplicación y medición de rendimiento de las técnicas de paralelismo propuestas.

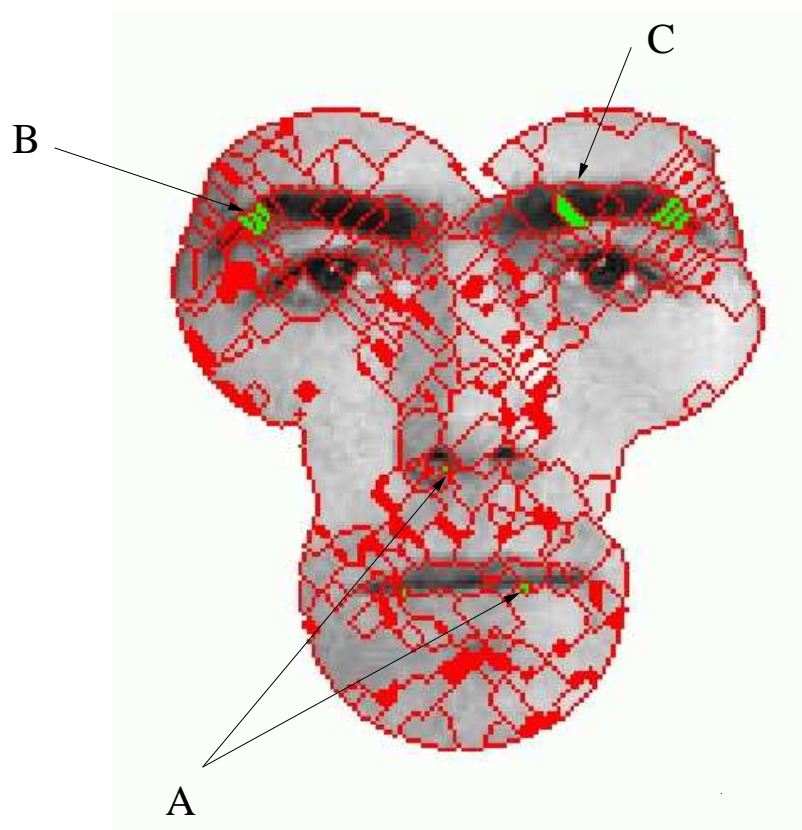


Figure 8: Ejemplo de algunos errores típicos en regiones. A: segmentos muy pequeños, de tamaño tan pequeño que es difícil incluso para una persona identificarlas propiamente. B: segmentos situados en cercanía de límites entre dos o más regiones, tan cercanos que crean dudas para poder clasificarse adecuadamente. C: verdaderos errores de reconocimiento.

Además de todos estos estudios, se muestran en esta tesis aplicaciones reales, orientadas sobre todo a reconocimiento de estructuras cerebrales a partir de imágenes en 3D de resonancia magnética, y de reconocimiento de estructuras faciales (se trata de encontrar la nariz, boca, etc.). Los resultados conseguidos para este último problema para los algoritmos con *steady state* (un algoritmo genético), UMDA y EBNA se muestran en la Figura 7.

En todos los ejemplos reales que se han utilizado en esta tesis se ha comprobado que los posibles errores de reconocimiento al utilizar técnicas de correspondencia entre grafos se dividen en tres tipos. Estos tres tipos se ilustran en la Figura 8.

Conclusiones y perspectivas

Esta tesis plantea los problemas de correspondencia entre grafos como problemas de optimización combinatorial con restricciones. Una de las aportaciones más novedosas de esta tesis consiste en utilizar EDAs por primera vez en este tipo de problemas.

De la comparación de EDAs con GAs la conclusión de los experimentos realizados es la siguiente: en problemas no muy complejos los GAs encuentran resultados similares a los de los EDA requiriendo menos tiempo de ejecución; sin embargo, en problemas complejos los EDAs siempre consiguen mejores resultados que los GAs, siendo estos últimos además muy

susceptibles a caer en máximos locales. Además, los EDA continuos muestran generalmente un mejor rendimiento que los discretos a la hora de devolver una solución, aunque a costa de un mayor tiempo de ejecución.

Se proponen asimismo varios tipos diferentes de funciones objetivo basados en diferentes paradigmas como por ejemplo lógica difusa y teoría de la probabilidad. Los resultados muestran también la importancia de la generación del grafo modelo y la definición de atributos.

Finalmente, los resultados experimentales muestran que la paralelización de los EDA contribuye satisfactoriamente a la reducción de tiempos de cálculo.

Referente a líneas de trabajo futura, se pueden mencionar las siguientes áreas y posibles ideas:

- * **Modelización:** considerar importancias diferentes entre diferentes regiones, generar modelos a partir de más de una imagen. . .
- * **Funciones objetivo:** realizar una comparación de rendimiento entre ellas, pruebas con otras representaciones de individuos y funciones objetivo. . .
- * **Comprobar la validez del método en secuencias de imágenes**
- * **Mejoras en los EDA:** otras técnicas de generación de poblaciones iniciales, pruebas con otros modelos gráficos probabilísticos. . .
- * **Paralelización de EDAs:** paralelizar otros EDAs, combinación entre algoritmos en paralelo, aplicar otras técnicas de paralelismo. . .

Publicaciones de esta tesis doctoral

Durante el desarrollo de esta tesis doctoral se han publicado varios trabajos en revistas, libros y congresos, sobre todo a nivel internacional. La lista completa de publicaciones es la siguiente:

2002

- Bengoetxea, E., Larrañaga, P., Bloch, I., Perchant, A., and Boeres, C. (2002). Learning and simulation of Bayesian networks applied to inexact graph matching. *Pattern Recognition* 35(12):2867-2880.
- Cesar R., Bengoetxea E., Bloch I. Inexact graph matching using stochastic optimization techniques for facial feature recognition. In *International Conference on Pattern Recognition (ICPR 2002)*. Quebeck, Canada.
- Cesar R., Bengoetxea E., Bloch I., Larrañaga, P. Inexact graph matching for facial feature recognition. *International Journal of Imaging Systems and Technology* (submitted).
- Mendiburu A., Bengoetxea, E., Miguel J. Paralelización de algoritmos de estimación de distribuciones. In *XIII Jornadas de Paralelismo*, pages 37-41. ISBN: 64-8409-159-7.

2001

- Bengoetxea, E., Larrañaga, P., Bloch, I., and Perchant, A. (2001). Estimation of Distribution Algorithms: A New Evolutionary Computation Approach For Graph Matching Problems. Lecture notes in Computer Science 2134. Third International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR-2001). M. Figueiredo, J. Zerubia and A. K. Jain (eds.), pages 454-468, Sophia Antipolis, France.
- Bengoetxea, E., Larrañaga, P., Bloch, I., and Perchant, A. (2001). Image Recognition with Graph Matching Using Estimation of Distribution Algorithms. Proceedings of the Medical Image Understanding and Analysis (MIUA 2001), 89-92, Birmingham, UK.
- Bengoetxea, E., Larrañaga, P., Bloch, I., Perchant, P. (2001). Solving graph matching with EDAs using a permutation-based representation. Estimation of Distribution Algorithms. A new tool for Evolutionary Computation, Larrañaga, P. and Lozano, J.A. (eds.). Kluwer Academic Publishers.
- Larrañaga, P., Bengoetxea, E., Lozano, J.A., Robles, V., Mendiburu, A., and de Miguel, P. (2001). Searching for the Best Permutation with estimation of Distribution Algorithms. Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI 2001). Workshop on Stochastic Search Algorithms, pages 7-14, Seattle, USA.
- Bengoetxea, E., Miquélez, T., Lozano, J. A., and Larrañaga, P. (2001). Experimental results in function optimization with EDAs in continuous domain. Estimation of Distribution Algorithms. A new tool for Evolutionary Computation, Larrañaga, P. and Lozano, J.A. (eds.). Kluwer Academic Publishers.
- Larrañaga, P., Lozano, J. A., and Bengoetxea, E. (2001). Estimation of Distribution Algorithms based on multivariate normal and Gaussian networks. Technical Report KZZA-IK-1-01 of the Department of Computer Science and Artificial Intelligence, University of the Basque Country, Spain.
- Miquélez, T., Bengoetxea, E., Morlán, I., and Larrañaga, P. (2001). Obtention of filters for Image Restoration Using Estimation of Distribution Algorithms. CAEPIA 2001, Spanish Society for the Artificial Intelligence. (in Spanish).

2000

- Bengoetxea, E., Larrañaga, P., Bloch, I., Perchant, A., and Boeres, C. (2000). Inexact graph matching using learning and simulation of Bayesian networks. An empirical comparison between different approaches with synthetic data. Proceedings of CaNew workshop, ECAI 2000 Conference, ECCAI. Berlin, Germany.

Résumé

Motivation

L'objectif de cette thèse est de contribuer au développement de méthodes de reconnaissance d'objets dans les images, s'appuyant sur des informations de haut niveau, sur la structure de la scène (topologie, organisation spatiale des différents objets dans l'image).

Au cours des dernières années les graphes ont été utilisés pour représenter mathématiquement la connaissance structurelle. L'intérêt de ce type de représentation va en augmentant dans la communauté scientifique grâce à la possibilité d'utiliser cette représentation en combinaison avec des algorithmes de mise en correspondance de graphes (*graph matching* en anglais), car cette représentation permet de représenter les variations et les différences structurelles entre objets différents.

Différentes applications ont été proposées en utilisant ce type de représentation pour réaliser la reconnaissance automatique des images. Dans ce type d'applications, la connaissance sur les variations structurelles possibles des objets qui apparaissent dans les images (personne, visage, image médicale, etc.) est exprimée à travers un modèle que l'on représente comme un graphe (ou atlas, comme on l'appelle dans certains cas). La cartographie, la reconnaissance de caractères, et la reconnaissance de structures cérébrales à partir d'images de résonance magnétique sont des exemples de domaines dans lesquels ce type de représentation a été utilisé et publié. Ce dernier exemple est illustré figure 9.

Le *graphe modèle* est souvent construit en utilisant un nœud pour chacune des régions de l'objet à reconnaître (Ex: dans le cas d'images d'un cerveau humain, il y aura un nœud pour représenter le cervelet, un autre pour chaque ventricule...) et les arêtes pour représenter les relations entre ces régions. On utilise des attributs pour exprimer les propriétés de chaque nœud ou arête, et pour pouvoir les identifier ainsi que les distinguer les uns des autres. Ce graphe est souvent construit de manière supervisée.

Après la construction d'un graphe modèle, et pour pouvoir réaliser la reconnaissance d'images à travers lui, il est nécessaire de construire un graphe à partir de chacune des images à reconnaître. Ces graphes, qui sont dénommés dans la littérature *graphes de données* ou *graphes d'entrée*, sont automatiquement construits par l'ordinateur sans l'assistance de l'utilisateur. Dans ce processus de construction du graphe de données correspondant à une image, une étape de segmentation de l'image précède la construction du graphe et est fondamentale : le graphe de données est formée en utilisant un nœud pour représenter chacun des segments obtenus. Les mêmes attributs que ceux du graphe modèle sont calculés pour les nœuds et les arêtes du graphe de données.

Mise en correspondance de graphes

La reconnaissance des structures de l'image est réalisée en recherchant la meilleure correspondance entre les graphes modèle et de données. Il s'agit d'affecter une étiquette à chacune des régions de l'image à reconnaître, les étiquettes étant définies par les régions du graphe modèle. La figure 10 illustre le principe de la mise en correspondance.

En notant $G_M = (V_M, E_M)$ le graphe modèle et $G_D = (V_D, E_D)$ le graphe de données, le problème consiste à trouver un homomorphisme $h : V_D \rightarrow V_M$ tel que $(u, v) \in E_D$ si et seulement si $(f(u), f(v)) \in E_M$. Ces homomorphismes sont évalués en calculant la similarité entre les attributs des nœuds et des arêtes mis en correspondance. On cherche alors un meilleur homomorphisme au sens d'un certain critère combinant ces similarités.

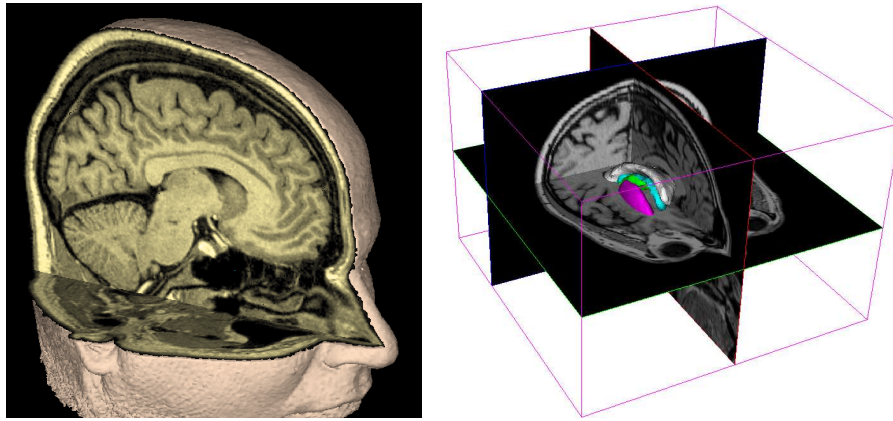


Figure 9: Exemple d'application de la mise en correspondance de graphes appliquée à la reconnaissance d'images. L'objectif est de reconnaître les différentes parties d'un cerveau humain à partir d'une image 3D comme celle de gauche.

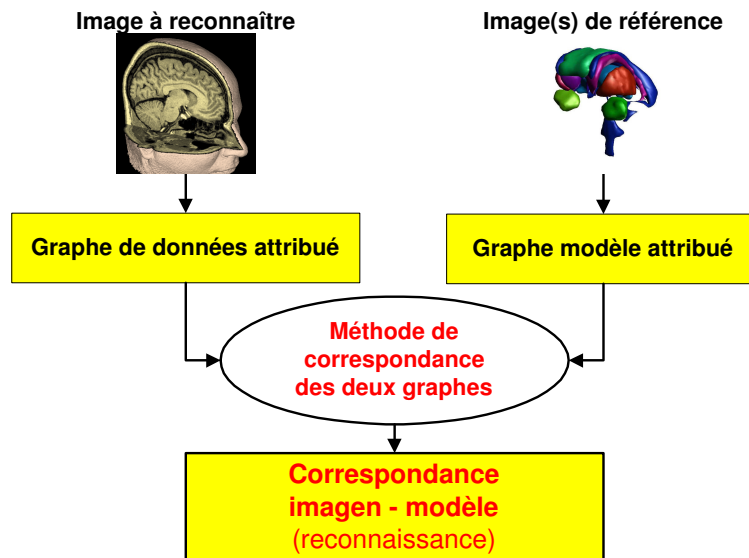


Figure 10: Principe de la mise en correspondance des graphes modèle et de données pour la reconnaissance de structures cérébrales.

Une grande partie de la littérature est consacrée à la recherche d'isomorphismes de graphes, ce qui suppose en particulier que $|V_M| = |V_D|$. On parle alors de correspondance exacte. Cependant, dans beaucoup de problèmes, surtout pour ceux où la segmentation s'effectue automatiquement, la condition d'isomorphisme est trop stricte et le nombre de nœuds du graphe de données est supérieur à celui du modèle ($|V_M| < |V_D|$). On parle alors de *correspondance inexacte de graphes*, et la complexité de ces problèmes est supérieure à celle des précédents. C'est à ce type de problème que nous nous intéressons ici.

La figure 11 illustre les différents types de problèmes de mise en correspondance de graphes. Dans les problèmes de mise en correspondance inexacte, on utilise souvent :

- un nœud nul ou *dummy* représentant la non-reconnaissance (par exemple pour une

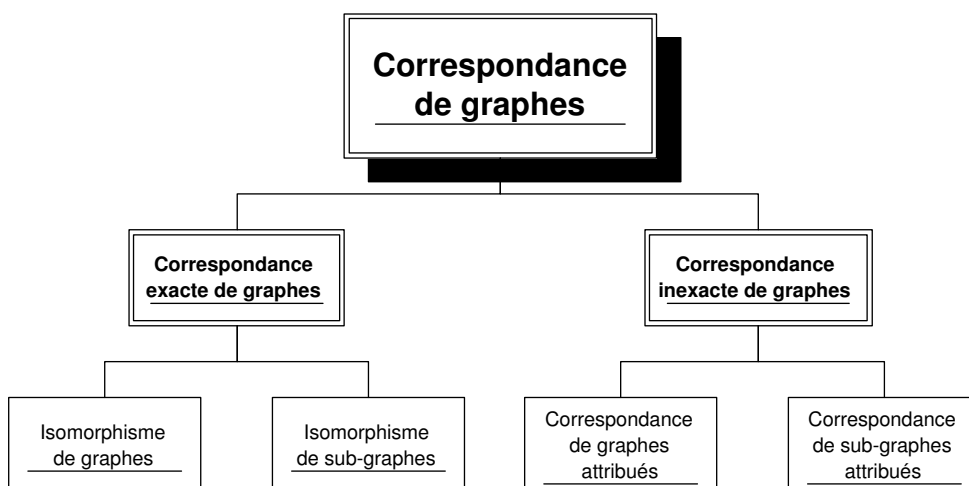


Figure 11: Classement des problèmes de mise en correspondance de graphes en deux classes principales.

région de l'image correspondant à une structure non représentée dans le modèle) ;

- des correspondances multiples entre nœuds : ces types de problèmes sont les plus complexes, car il est possible qu'un segment de l'image à reconnaître fasse en même temps partie de plus d'une région de l'objet à traiter. Bien que ce type de problèmes puisse exister, leur résolution implique plus de combinaisons et la recherche pour trouver la solution optimale peut être de complexité trop élevée pour pouvoir être traitée.

Beaucoup de techniques très différentes ont été développées pour la mise en correspondance de graphes : arbres de décision [Messmer and Bunke, 1999], réseaux de neurones [Riviere et al., 2002], algorithme EM [Cross and Hancock, 1998], relaxation probabiliste [Christmas et al., 1995], heuristiques et métaheuristiques [Pelillo et al., 1999], algorithmes génétiques [Wilson and Hancock, 1997], et programmation évolutive [Singh and Chaudhury, 1997]. Beaucoup de ces techniques restent restreintes aux correspondances exactes de graphes. Cependant, dans beaucoup d'autres exemples, surtout en ce qui concerne les applications aux images réelles, la condition d'isomorphisme est trop forte, cependant les approches pour résoudre les problèmes de correspondances inexactes sont actuellement moins répandues.

Les problèmes considérés dans cette thèse sont des problèmes de mise en correspondance inexacte de graphes, sans ajouter de nœud nul. En outre, dans le but de ne pas augmenter la complexité de problèmes qui sont déjà complexes, il a été décidé de réaliser une sursegmentation de l'image à reconnaître pour éviter la possibilité de correspondances multiples entre graphes. La méthode proposée pour réaliser la mise en correspondance repose sur les algorithmes d'estimation de distributions (EDAs).

Formalisation de la mise en correspondance de graphes comme un problème d'optimisation combinatoire

A cause de la grande complexité du problème de mise en correspondance inexacte, nous nous sommes tournés vers des algorithmes d'optimisation combinatoire sous contrainte. Cette thèse analyse les aspects et mécanismes qui doivent s'appliquer pour pouvoir formuler le

problème de cette manière et ensuite appliquer des algorithmes tels que les Algorithmes d'Estimation de Distributions (EDAs).

Les caractéristiques suivantes doivent être définies :

- **une représentation des individus** ou des solutions, de manière à représenter chacun des points dans l'espace de recherche ;
- **une fonction objectif à optimiser** qui affecte une valeur à chaque solution possible ou individu, pour exprimer la validité et la qualité de l'individu comme solution au problème.

Chaque individu est composé d'un vecteur de valeurs qui peuvent être discrètes ou continues. Les EDAs permettent d'utiliser les deux types d'individus contrairement à d'autres algorithmes. Cette thèse présente la façon de travailler avec les deux types d'individus et de représentations afin de les appliquer dans des problèmes de mise en correspondance de graphes.

Une représentation possible des individus consiste à associer à chaque nœud de G_D un nœud de G_M . Donc, la taille des individus est déterminée dans ce cas par $n = |V_D|$ variables⁴, $X_i \in \mathbf{X}$ $i = 1, \dots, |V_D|$, où chaque variable contient une valeur entre 1 et $|V_M|$. La valeur de chaque variable dans l'individu a la signification suivante : $X_i = k$ avec $1 \leq i \leq |V_D|$ et $1 \leq k \leq |V_M| \Leftrightarrow$ le i -ème nœud de G_D est identifié comme le k -ème nœud du graphe modèle G_M .

Cette thèse propose deux autres représentations d'individus dans le domaine discret, ainsi qu'une représentation pour le domaine continu. En outre, et avec l'intention de montrer la robustesse de la méthode proposée, une contrainte supplémentaire que doivent accomplir les individus pour pouvoir être pris en considération est ajoutée :

$$\forall a_M \in V_M, \exists a_D \in V_D \mid h(a_D) = a_M$$

On impose ainsi que toute structure du modèle soit identifiée dans l'image. Cette restriction augmente ainsi la complexité des problèmes et celle des différents mécanismes présentés dans cette thèse. Elle permet de montrer comment on peut tenir compte des restrictions quand on applique les EDAs.

Enfin, dans le but d'évaluer la qualité de l'individu comme solution possible à un problème, on définit deux fonctions $c_N(a_D, a_M)$ et $c_E(e_D, e_M)$ qui mesurent respectivement la similarité entre deux nœuds $a_D \in V_D$ et $a_M \in V_M$, et entre deux arêtes $e_D \in E_D$ et $e_M \in E_M$. Nous avons défini plusieurs fonctions objectif dépendant de ces fonctions de similarité.

Algorithmes d'Estimation de Distributions

Les Algorithmes d'Estimation de Distributions (EDAs) sont une approche récente dans la famille des algorithmes évolutifs appliqués aux problèmes d'optimisation. Ces algorithmes se caractérisent par un procédé d'évolution d'un ensemble de solutions. À cause de la similitude avec des populations d'êtres vivants, on appelle d'habitude *individu* chacune des solutions, *population* l'ensemble d'individus, et *génération* chacune des populations qui se développent successivement. La différence la plus significative entre les EDAs et les Algorithmes Génétiques (GAs), qui sont parmi les plus connus et utilisés dans ce domaine, réside

⁴Dans le domaine des algorithmes génétiques on considère que les individus sont composés de gènes, mais dans le domaine des EDAs on parle plutôt de variables probabilistes.

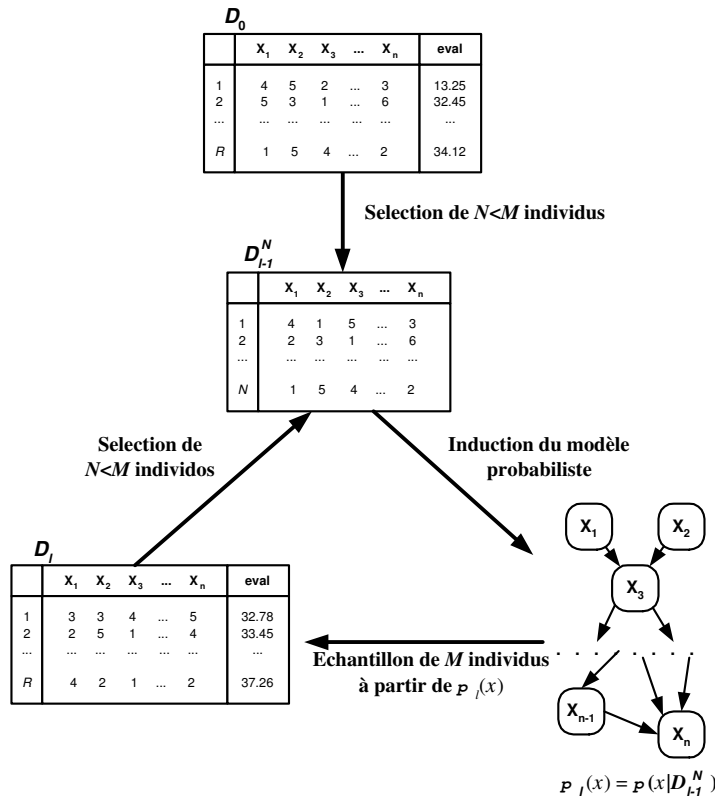


Figure 12: Illustration de l'approche EDA en optimisation.

dans le processus d'évolution d'une génération à la suivante : elle est réalisée dans le cas des GAs par des opérations de mutation et de croisement, et dans les EDAs par des techniques fondées sur la théorie des probabilités et plus particulièrement sur l'apprentissage et la simulation de réseaux bayésiens ou de réseaux gaussiens. Cette idée est illustrée figure 12.

La figure 13 présente le pseudocode générique des EDAs qui suit essentiellement les étapes suivantes :

1. La population initiale D_0 formée par R individus est créée. La génération de ces R individus est souvent réalisée en supposant une distribution uniforme pour chaque variable. Une fois que les individus sont engendrés, ils sont évalués selon la fonction objectif.
2. Pour faire évoluer la i -ème population D_{l-1} vers la suivante D_l , N individus sont sélectionnés dans D_{l-1} ($N < R$) en suivant un certain critère. Nous appelons D_{l-1}^N l'ensemble des N individus sélectionnés dans la génération numérotée $l - 1$.
3. Le modèle probabiliste graphique n -dimensionnel qui représente le mieux les dépendances entre les n variables est déterminé. C'est l'étape d'*apprentissage*, qui est cruciale dans les EDAs à cause de l'importance de la prise en compte de toutes les dépendances entre les variables pour assurer une évolution satisfaisante vers des individus meilleurs.
4. La nouvelle population D_l est constituée avec R nouveaux individus obtenus après avoir simulé la distribution de probabilité apprise dans l'étape précédente. On utilise souvent une approche élitiste, et ainsi le meilleur individu de la population D_{l-1}^N est

EDA

$D_0 \leftarrow$ Engendrer R individus (la population initiale D_0) au hasard

Répéter pour $l = 1, 2, \dots$ jusqu'à satisfaire un critère d'arrêt

$D_{l-1}^N \leftarrow S$ $N < R$ individus de D_{l-1} en suivant
une méthode de sélection déterminée

$\rho_l(\mathbf{x}) = \rho(\mathbf{x} | D_{l-1}^N) \leftarrow$ Estimer la distribution de probabilité
qu'un individu se trouve entre les individus sélectionnés

$D_l \leftarrow$ Echantillonner R individus, la nouvelle population, à partir de $\rho_l(\mathbf{x})$

Figure 13: Pseudocode générique des EDAs.

gardé dans la nouvelle population D_l . De cette manière, dans chaque génération un total de $R - 1$ nouveaux individus sont créés au lieu de R .

Les étapes 2, 3 et 4 sont répétées jusqu'à satisfaire une condition d'arrêt. Des exemples de critères d'arrêt sont : arriver à un nombre de génération maximal, atteindre un nombre maximal d'individus analysés, uniformité dans la population, ou le fait de ne pas obtenir un individu avec une valeur de fonction objectif meilleure après un certain nombre de générations.

Un grand nombre d'algorithmes qui font partie des EDAs a été proposé dans la littérature. On peut les classer en trois grands groupes selon la complexité du type de dépendances entre variables considéré :

- **Sans dépendance entre variables** : la structure du réseau bayésien (ou gaussien si nous travaillons dans le domaine continu) est fixe et ne contient pas d'arcs. En d'autres termes, cela signifie que toutes les variables de cet individu sont considérées indépendantes entre elles.

Comme exemple d'algorithmes du domaine discret qui appartiennent à ce groupe nous pouvons mentionner UMDA (Univariate Marginal Distribution Algorithm) [Mühlenbein, 1998] où l'estimation de la distribution de probabilité est réalisée de la manière suivante :

$$p(\mathbf{x}) = \prod_{i=1}^n p(x_i)$$

Un autre exemple est l'algorithme connu comme UMDA_c (Univariate Marginal Distribution Algorithm - continuous) [Larrañaga et al., 2000], qui est l'équivalent de l'UMDA dans le domaine continu.

- **Dépendances entre paires de variables** : dans la structure du réseau bayésien ou gaussien chacune des variables peut avoir au plus un parent. Cela nécessite donc une étape d'estimation de la meilleure structure.

Un exemple d'EDAs pour le domaine discret appartenant à ce groupe est MIMIC (Mutuel Information Maximization for Input Clustering) [de Bonet et al., 1997] qui propose de réaliser la factorisation de probabilité suivante :

$$p(\mathbf{x}) = p(x_{i_1} | x_{i_2}) \cdot p(x_{i_2} | x_{i_3}) \cdots p(x_{i_{n-1}} | x_{i_n}) \cdot p(x_{i_n})$$

MIMIC recherche la permutation $\pi = (i_1, i_2, \dots, i_n)$ qui minimise la divergence de Kullback-Leibler entre la distribution estimée $\hat{p}_\pi(\mathbf{x})$ et la distribution réelle $p(\mathbf{x})$.

De nouveau, il existe une version continue de MIMIC nommée MIMIC_c (Mutuel Information Maximization for Input Clustering - continuous) [Larrañaga et al., 2000].

- **Dépendances multiples entre variables :** cette fois toutes les probabilités conditionnelles peuvent intervenir, et la structure du réseau bayésien ou gaussien est quelconque. Cette caractéristique demande une recherche exhaustive de la meilleure structure probabiliste graphique parmi toutes celles qui sont possibles, et donc ces algorithmes sont plus chers en termes de temps d'exécution que ceux des groupes précédents. Leur avantage est qu'ils sont capables d'apprendre une structure probabiliste graphique qui considère les relations entre les différentes variables des individus plus fidèlement.

Citons comme exemple dans le domaine discret l'algorithme EBNA (Estimation of Bayesian Networks Algorithm) [Etxeberria and Larrañaga, 1999], et EGNA (Estimation of Gaussian Networks Algorithm) [Larrañaga et al., 2000, Larrañaga and Lozano, 2001] dans le domaine continu.

L'algorithme EBNA utilise un *score* BIC (Bayesian Information Criterion) [Schwarz, 1978] qui mesure l'aptitude d'une structure à représenter les dépendances entre les individus. Le réseau bayésien qui maximise cette mesure est estimé par la méthode de l'Algorithme B [Buntine, 1991]. La factorisation de la probabilité s'écrit alors de la façon suivante :

$$p(\mathbf{x}) = \prod_{i=1}^n p(x_i | \mathbf{pa}(x_i))$$

où $\mathbf{pa}(x_i)$ est l'ensemble des parents de la variable x_i dans le réseau bayésien.

Les algorithmes et approches d'EDAs qui sont proposés dans la littérature sont nombreux, bien que pour le moment il y n'ait que peu d'articles qui en montrent tout le potentiel en comparaison avec d'autres paradigmes plus connus comme par exemple les GAs et les Stratégies Evolutives (ES). Cette thèse vise à combler ce manque dans le cas de la mise en correspondance de graphes pour la reconnaissance d'objets dans les images. Un autre des aspects innovants de cette thèse est précisément le fait d'appliquer pour la première fois les EDAs aux problèmes de mise en correspondance de graphes, tout en tenant compte des deux domaines discret et continu.

Dans ce cas, les distributions estimées dans les EDAs représentent les dépendances entre les différentes correspondances possibles entre nœuds de G_D par rapport aux nœuds de G_M .

EDAs parallèles

Une des principales limites des EDAs pour les problèmes traités ici est le temps de calcul important qu'ils nécessitent pour aboutir à la solution. Nous avons donc étudié les possibilités de parallélisation de ces algorithmes afin de réduire ces temps d'exécution.

Nous avons analysé différents EDAs en fonction des besoins de calcul pour chacune de leurs étapes. Cette analyse a été réalisée avec l'outil GNU *gprog*. Ces études soulignent que les algorithmes les plus chers en calcul sont ceux du troisième groupe, et cet effet est précisément dû à l'étape d'apprentissage. Dans le cas de la reconnaissance de structures

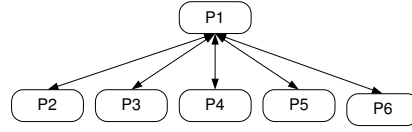


Figure 14: Schéma d'exécution maître-esclave, où un processus joue le rôle de *maître* de tâches et les autres réalisent des parties d'un travail commun.

cérébrales, cet apprentissage avec EBNA prend 85,7% du temps d'exécution total. Nous avons décidé de paralléliser l'algorithme EBNA (troisième groupe et domaine discret).

L'apprentissage dans EBNA repose sur la mesure BIC, qui demande presque tout le temps de calcul de cet apprentissage. La mesure $BIC(S, D)$, où S est la structure et D les données à partir desquelles est réalisé l'apprentissage, peut être divisée en composantes $BIC(i, S, D)$ qui expriment la mesure locale BIC pour une variable X_i :

$$BIC(S, D) = \sum_{i=1}^n BIC(i, S, D) = \sum_{i=1}^n \sum_{j=1}^{q_i} \sum_{k=1}^{r_i} N_{ijk} \log \frac{N_{ijk}}{N_{ij}} - \frac{1}{2}(r_i - 1)q_i$$

Pour paralléliser le programme on utilise le modèle d'exécution maître-esclave illustré figure 14, dans lequel chacun des esclaves calcule les termes $BIC(i, S, D)$ correspondant aux différentes variables X_i $i = 1, \dots, n$.

Pour essayer des techniques distinctes de parallélisme, cette thèse utilise des bibliothèques pthreads et MPI qui permettent de tester l'intérêt d'utiliser la *mémoire partagée* et le *passage des messages* respectivement, avec des configurations différentes des ordinateurs.

Les essais réalisés montrent qu'avec des threads et avec du MPI nous pouvons obtenir des réductions significatives du temps de calcul, allant jusqu'à 60%.

Résultats expérimentaux

Nous avons traité différents exemples d'application des EDAs à des problèmes de correspondance inexacte de graphes. Des exemples synthétiques obtenus par simulations permettent d'illustrer différentes caractéristiques des EDAs et des techniques proposées :

- Comparaison entre EDAs, algorithmes génétiques et stratégies évolutives pour des problèmes de complexités très différentes.
- Analyse de quatre méthodes adaptées aux EDAs pour satisfaire les contraintes imposées aux problèmes de correspondance de graphes.
- Etude de l'évolution des structures graphiques probabilistes de génération en génération pour les EDAs discrets ou continus.
- Application et mesure de rendement des techniques de parallélisme proposées.

Des applications réelles ont ensuite été traitées, portant sur la reconnaissance de structures cérébrales à partir d'images 3D de résonance magnétique, et de structures faciales (il s'agit de trouver le nez, la bouche, etc.). Les résultats obtenus pour ce dernier problème pour les algorithmes ssGA (algorithme génétique), UMDA et EBNA sont illustrés figure 15.

Dans tous les exemples réels traités, les erreurs de reconnaissance que peuvent commettre les techniques de correspondance entre graphes se divisent en trois types, illustrés figure 16.

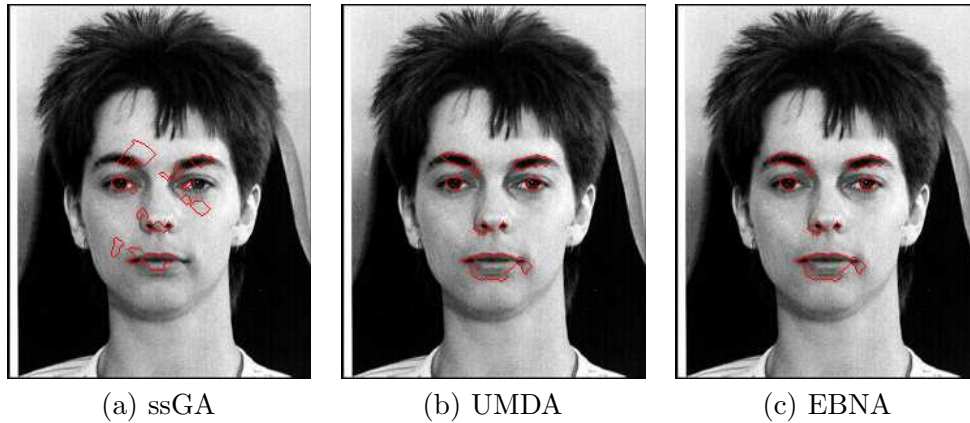


Figure 15: Exemple d'un problème réel de reconnaissance de structures faciales. Les résultats sont obtenus en utilisant l'algorithme génétique SSGA et les EDAs UMDA et EBNA. Ces résultats montrent que la reconnaissance est meilleure dans le cas des EDAs.

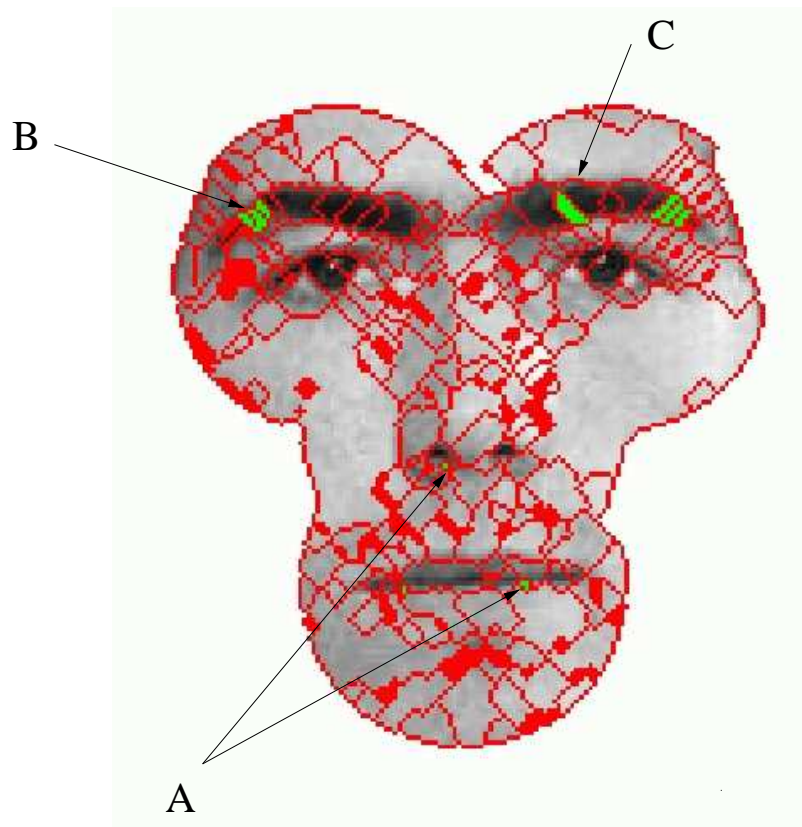


Figure 16: Exemple de quelques erreurs typiques de reconnaissance. A: segments de taille trop petite pour pouvoir être identifiés, même visuellement. B: segments situés à proximité des limites de deux régions ou plus, trop ambigus pour pouvoir être classés. C: vraies erreurs de reconnaissance.

Conclusions et perspectives

Cette thèse formalise les problèmes de mise en correspondance de graphes comme des problèmes d'optimisation combinatoire sous contraintes. Un des apports les plus innovants de cette thèse consiste à utiliser pour la première fois les EDAs dans ce type de problèmes.

Quant à la comparaison des EDAs avec des GAs, la conclusion des essais réalisés est la suivante : pour des problèmes qui ne sont pas très complexes les GAs donnent des résultats similaires à ceux des EDAs et nécessitent moins de temps d'exécution ; cependant, pour des problèmes complexes les EDAs donnent toujours de meilleurs résultats que les GAs, qui sont plus susceptibles de tomber dans des extrema locaux. En outre, les EDAs du domaine continu se comportent généralement mieux que ceux du domaine discret quant à la qualité de la solution obtenue, bien qu'avec un coût plus grand en termes de temps d'exécution.

Différents types de fonctions objectif ont également été proposés, fondés sur des paradigmes différents comme la théorie des ensembles flous et la théorie des probabilités. Les résultats montrent aussi l'importance de choisir une bonne façon de générer le graphe modèle et de définir les attributs.

Finalement, les résultats expérimentaux montrent que la parallélisation des EDAs contribue de façon satisfaisante à la réduction de temps de calcul.

Les perspectives de ce travail sont les suivantes :

Modélisation : considérer les différences d'importance entre les régions, engendrer des modèles à partir de plus d'une image.

Fonctions objectif : réaliser une comparaison de rendement entre elles, tester d'autres représentations d'individus et de fonctions objectif.

Vérifier la validité de la méthode sur des séquences d'images et exploiter la continuité d'une image à la suivante pour améliorer la reconnaissance.

Amélioration des EDAs : autres techniques de génération de populations initiales, essais avec d'autres modèles graphiques probabilistes.

Parallélisation des EDAs : paralléliser d'autres EDAs, combinaison entre algorithmes en parallèle, appliquer d'autres techniques de parallélisme.

Publications des travaux réalisés pendant la thèse

Cette thèse a donné lieu à plusieurs publications dans des revues et congrès internationaux.

2002

- Bengoetxea, E., Larrañaga, P., Bloch, I., Perchant, A., and Boeres, C. (2002). Learning and simulation of Bayesian networks applied to inexact graph matching. *Pattern Recognition* 35(12):2867-2880.
- Cesar R., Bengoetxea E., Bloch I. Inexact graph matching using stochastic optimization techniques for facial feature recognition. In *International Conference on Pattern Recognition (ICPR 2002)*. Quebec, Canada.
- Cesar R., Bengoetxea E., Bloch I., Larrañaga, P. Inexact graph matching for facial feature recognition. *International Journal of Imaging Systems and Technology* (submitted).

-
- Mendiburu A., Bengoetxea, E., Miguel J. Paralelización de algoritmos de estimación de distribuciones. In XIII Jornadas de Paralelismo, pages 37-41. ISBN: 64-8409-159-7.

2001

- Bengoetxea, E., Larrañaga, P., Bloch, I., and Perchant, A. (2001). Estimation of Distribution Algorithms: A New Evolutionary Computation Approach For Graph Matching Problems. Lecture notes in Computer Science 2134. Third International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR-2001). M. Figueiredo, J. Zerubia and A. K. Jain (eds.), pages 454-468, Sophia Antipolis, France.
- Bengoetxea, E., Larrañaga, P., Bloch, I., and Perchant, A. (2001). Image Recognition with Graph Matching Using Estimation of Distribution Algorithms. Proceedings of the Medical Image Understanding and Analysis (MIUA 2001), 89-92, Birmingham, UK.
- Bengoetxea, E., Larrañaga, P., Bloch, I., Perchant, P. (2001). Solving graph matching with EDAs using a permutation-based representation. Estimation of Distribution Algorithms. A new tool for Evolutionary Computation, Larrañaga, P. and Lozano, J.A. (eds.). Kluwer Academic Publishers.
- Larrañaga, P., Bengoetxea, E., Lozano, J.A., Robles, V., Mendiburu, A., and de Miguel, P. (2001). Searching for the Best Permutation with estimation of Distribution Algorithms. Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI 2001). Workshop on Stochastic Search Algorithms, pages 7-14, Seattle, USA.
- Bengoetxea, E., Miquélez, T., Lozano, J. A., and Larrañaga, P. (2001). Experimental results in function optimization with EDAs in continuous domain. Estimation of Distribution Algorithms. A new tool for Evolutionary Computation, Larrañaga, P. and Lozano, J.A. (eds.). Kluwer Academic Publishers.
- Larrañaga, P., Lozano, J. A., and Bengoetxea, E. (2001). Estimation of Distribution Algorithms based on multivariate normal and Gaussian networks. Technical Report KZZA-IK-1-01 of the Department of Computer Science and Artificial Intelligence, University of the Basque Country, Spain.
- Miquélez, T., Bengoetxea, E., Morlán, I., and Larrañaga, P. (2001). Obtention of filters for Image Restoration Using Estimation of Distribution Algorithms. CAEPIA 2001, Spanish Society for the Artificial Intelligence. (in Spanish).

2000

- Bengoetxea, E., Larrañaga, P., Bloch, I., Perchant, A., and Boeres, C. (2000). Inexact graph matching using learning and simulation of Bayesian networks. An empirical comparison between different approaches with synthetic data. Proceedings of CaNew workshop, ECAI 2000 Conference, ECCAI. Berlin, Germany.

Table of Contents

1	Introduction	1
2	The graph matching problem	3
2.1	Basic notation and terminology	3
2.2	Definition and classification of graph matching problems	4
2.2.1	Exact and inexact graph matching	5
2.2.2	Graph matching using dummy vertices	6
2.2.3	Graph matching allowing more than one correspondence per vertex	7
2.3	Complexity of graph matching	7
2.3.1	Exact graph matching: graph isomorphism	8
2.3.2	Exact sub-graph matching: sub-graph isomorphism	8
2.3.3	Inexact graph matching: graph and sub-graph homomorphisms	8
2.4	State of the art in the literature	8
2.4.1	Image processing and graph construction techniques for graph matching	9
2.4.2	Distance measures, conceptual graphs, and graph edit distances and metrics	11
2.4.3	Graph matching using genetic algorithms	13
2.4.4	Graph matching using techniques based on probability theory	14
2.4.5	Applying decision trees to graph matching	15
2.4.6	Graph matching using neural networks	16
2.4.7	Graph matching using clustering techniques	16
2.4.8	Discussion	16
2.5	Graph matching problem types selected for this thesis	17
3	Graph matching as a combinatorial optimization problem with constraints	19
3.1	Introduction	19
3.2	Graph matching problems with special constraints in the literature	20
3.3	Representations of graph matching solutions by means of individuals	20
3.3.1	Individual representations in the discrete domain	21
3.3.2	Individual representations in the continuous domain	24
3.3.3	Conditions for a correct individual in a graph matching problem	25
3.3.4	What to do with incorrect individuals?	28
3.4	Fitness functions applied to graph matching	28
3.4.1	Graph attributes and similarities	29
3.4.2	f_1 : only taking into account the matched vertices	29
3.4.3	f_2 : taking into account all the vertices and similarities	30
3.4.4	f_3 : ignoring internal edges of vertices matched to the same model vertex	30

TABLE OF CONTENTS

3.4.5	f_4 : function based on the divergence between distributions	31
3.4.6	f_5 : function based on likelihood	39
3.5	Conclusion	41
4	Estimation of distribution algorithms	43
4.1	Introduction	43
4.2	Probabilistic graphical models	46
4.2.1	Bayesian networks	46
4.2.2	Simulation in Bayesian networks	47
4.2.3	Gaussian networks	48
4.2.4	Simulation in Gaussian networks	50
4.3	Estimation of distribution algorithms in discrete domains	50
4.3.1	Introduction	50
4.3.2	Without interdependencies	51
4.3.3	Pairwise dependencies	52
4.3.4	Multiple interdependencies	54
4.4	Estimation of distribution algorithms in continuous domains	58
4.4.1	Introduction	58
4.4.2	Without dependencies	59
4.4.3	Bivariate dependencies	60
4.4.4	Multiple interdependencies	61
4.5	Estimation of distribution algorithms for inexact graph matching	68
4.5.1	Discrete domains	68
4.5.2	Continuous domains	74
5	Parallel estimation of distribution algorithms	77
5.1	Introduction	77
5.2	Sequential programs and parallelization	78
5.2.1	The design of parallel programs	78
5.2.2	Parallelizing an already existing sequential program	84
5.3	Parallel architectures and systems	85
5.3.1	Parallel computer architectures	85
5.3.2	Comparison of parallel programming models	88
5.3.3	Communication in distributed systems: existing libraries	89
5.4	Parallelism techniques in the literature applied to graph matching	90
5.5	Parallelization of sequential EDA programs	91
5.5.1	Computation needs in EDAs	91
5.5.2	Analysis of the execution times for the most important parts of the sequential EDA program	92
5.5.3	Interesting properties of the BIC score for parallelization	94
5.5.4	Parallel techniques applied in this thesis	95
6	Experiments with synthetic examples	97
6.1	Introduction	97
6.2	Study 1: measurement of the performance	97
6.2.1	Design of the experiment	97
6.2.2	The need to obtain correct individuals	99
6.2.3	Discrete domain	102

6.2.4	Continuous domain	106
6.3	Study 2: evolution of probabilistic graphical structures	110
6.4	Study 3: parallelization	115
6.4.1	Parallelizing EBNA using threads and shared memory	116
6.4.2	Parallelizing EBNA using processes and MPI	117
6.5	Conclusions of the studies on synthetic problems	120
7	Experiments with real examples	125
7.1	Introduction	125
7.2	Recognition of brain images	125
7.2.1	Motivation	125
7.2.2	Construction of the atlas and data graphs	126
7.2.3	Description of the problem and the graph matching approach	127
7.2.4	Experimental results	129
7.2.5	Computational complexity and parallelization of the problem	131
7.3	Recognition of human facial features	133
7.3.1	Motivation	133
7.3.2	Construction of the model and data graphs	133
7.3.3	Description of two face feature recognition problems and the graph matching approach	137
7.3.4	Experimental results	139
7.4	Conclusions of the experiments on real problems	145
8	Conclusions and future work	149
8.1	Conclusions	149
8.2	Future work	150
A	Example on how to use a permutation-based approach in EDAs	153
A.1	Example of translating from a permutation to the solution it symbolizes	153
A.2	The redundancy problem on permutation-based representations	156
B	Example of correcting individuals	157
B.1	Simulation with LTM	158
B.2	Simulation with ATM	159
C	On processes and threads: synchronization and communication in parallel programs	161
C.1	Sequential and parallel programs: main differences	161
C.2	Processes and threads	162
C.3	Communication and synchronization between processes or threads	163
C.3.1	The model of race conditions and critical sections	164
C.3.2	Exclusive access to critical sections	165
C.3.3	Conditions for critical sections	166
C.3.4	Communication primitives	167
C.3.5	Message passing	173
C.3.6	Communication and synchronization paradigms	175

D	Analysis and parallelization of the source code of EBNA_{BIC}	177
D.1	Short review of the source code of the sequential EDA program	177
D.2	Parallelization using threads	179
D.2.1	New adaptation on the source code for threads	179
D.3	Parallelization using MPI	187
D.3.1	New adaptation on the source code for MPI	188

List of Figures

1	Ejemplo de aplicación de la correspondencia entre grafos al reconocimiento de imágenes. El objetivo en este problema en concreto es el de identificar las diferentes partes del cerebro a partir de la imagen en 3D de la izquierda.	vi
2	Ejemplo de cómo se realiza el reconocimiento de imágenes a partir de los grafos modelo y de datos.	vii
3	Clasificación de problemas de correspondencias entre grafos en dos clases principales.	viii
4	Ilustración de la aproximación EDA en procesos de optimización.	x
5	Pseudocódigo genérico de los EDA.	xi
6	Esquema de ejecución maestro-esclavo, donde un proceso hace de <i>gestor</i> de tareas y los demás realizan partes de un trabajo común.	xiii
7	Ejemplo de un problema real de reconocimiento de estructuras faciales utilizando correspondencia entre grafos. Se ilustran los resultados conseguidos utilizando algoritmos genéticos y los EDAs UMDA y EBNA. Estos resultados muestran que el reconocimiento en este caso es superior en el caso de ambos EDAs.	xiv
8	Ejemplo de algunos errores típicos en regiones. A: segmentos muy pequeños, de tamaño tan pequeño que es difícil incluso para una persona identificarlas propiamente. B: segmentos situados en cercanía de límites entre dos o más regiones, tan cercanos que crean dudas para poder clasificarse adecuadamente. C: verdaderos errores de reconocimiento.	xv
9	Exemple d'application de la mise en correspondance de graphes appliquée à la reconnaissance d'images. L'objectif est de reconnaître les différentes parties d'un cerveau humain à partir d'une image 3D comme celle de gauche.	xx
10	Principe de la mise en correspondance des graphes modèle et de données pour la reconnaissance de structures cérébrales.	xx
11	Classement des problèmes de mise en correspondance de graphes en deux classes principales.	xxi
12	Illustration de l'approche EDA en optimisation.	xxiii
13	Pseudocode générique des EDAs.	xxiv
14	Schéma d'exécution maître-esclave, où un processus joue le rôle de <i>maître</i> de tâches et les autres réalisent des parties d'un travail commun.	xxvi
15	Exemple d'un problème réel de reconnaissance de structures faciales. Les résultats sont obtenus en utilisant l'algorithme génétique SSGA et les EDAs UMDA et EBNA. Ces résultats montrent que la reconnaissance est meilleure dans le cas des EDAs. . .	xxvii
16	Exemple de quelques erreurs typiques de reconnaissance. A: segments de taille trop petite pour pouvoir être identifiés, même visuellement. B: segments situés à proximité des limites de deux régions ou plus, trop ambigus pour pouvoir être classés. C: vraies erreurs de reconnaissance.	xxvii

LIST OF FIGURES

2.1	Illustration of how the physical parts of a human body can be represented using a graph.	4
2.2	Classification of all the graph matching types into two main classes: exact graph matching and inexact graph matching (in which the best among all the possible non necessarily bijective matchings has to be found).	6
3.1	Pseudocode to compute the solution represented by a permutation-based individual.	23
3.2	Pseudocode to translate from a continuous value in \mathfrak{R}^n to a discrete permutation composed of discrete values.	25
3.3	Representation of a correct individual (a), and an incorrect individual (b) for our graph matching problem example, for the case that G_M (the model graph) contains 8 vertices and G_D (the data graph representing the segmented image) contains 14 vertices.	27
3.4	Illustration of two regions in the model graph (left), and the typical result after following an over-segmentation process on an image to be recognized (right). This figure also illustrates the centers of gravity of each of the regions. These will be used as a destination point representative of the whole vector from any point of the origin to the destination.	34
3.5	Example showing how the edge attributes are computed in both the model and data graphs.	34
3.6	Summary of the problem of obtaining the edge attributes between regions A and B in the model graph, knowing the values of the edge attributes in the data graph. . .	35
4.1	Illustration of EDA approaches in the optimization process.	45
4.2	Structure, local probabilities and resulting factorization in a Bayesian network for four variables (with X_1, X_3 and X_4 having two possible values, and X_2 with three possible values).	47
4.3	Pseudocode for the Probabilistic Logic Sampling method.	48
4.4	Structure, local densities and resulting factorization for a Gaussian network with four variables.	49
4.5	Pseudocode for EDA approaches in discrete domains.	51
4.6	Graphical representation of proposed EDA in combinatorial optimization with pairwise dependencies (MIMIC, tree structure, BMDA).	53
4.7	MIMIC approach to estimate the mass joint probability distribution.	54
4.8	Graphical representation of proposed EDA in combinatorial optimization with multiply dependencies (FDA, EBNA, BOA and EcGA).	55
4.9	Pseudocode for EBNA _{BIC} algorithm.	56
4.10	Pseudocode for EBNA _{K2} algorithm.	57
4.11	Pseudocode to estimate the joint density function followed in UMDA _c	60
4.12	Adaptation of the MIMIC approach to a multivariate Gaussian density function. . .	61
4.13	Pseudocode for the EMNA _{global} approach.	62
4.14	Pseudocode for the EMNA _a approach.	63
4.15	Pseudocode for the EGNA _{ee} , EGNA _{BGe} , and EGNA _{BIC} algorithms.	64
4.16	Pseudocode for Last Time Manipulation (LTM).	71
4.17	Pseudocode for All Time Manipulation (ATM).	73

5.1	Parallel design methodology when parallelizing programs, as described in [Foster, 1995]. The four stages are illustrated: starting from a problem specification, (1) the problem is partitioned in smaller tasks that are divided in processes, (2) communication requirements between the different processes are determined, (3) processes are agglomerated, and finally (4) processes are mapped to processors.	79
5.2	The producer-consumer approach.	80
5.3	The phase parallel approach.	82
5.4	The divide and conquer approach.	82
5.5	The pipeline approach.	82
5.6	The master-slave approach.	83
5.7	The work pool approach.	83
5.8	Illustration of all the different computer architectures. Here all the possible forms are shown, and many of them exist today. In some of these systems the number of processors is just one, but in other there can be thousands of them.	86
5.9	Illustration of the most used memory models for parallel computer architecture taxonomy.	87
6.1	Figures for the correctness of the UMDA, MIMIC and EBNA (discrete EDAs) as well as for the cGA, eGA and ssGA (GAs), applied to the second example of 30 & 100 vertices without correction.	100
6.2	Figures for the correctness of the UMDA, MIMIC and EBNA discrete EDAs as well as for the cGA, eGA and ssGA GAs, applied to the second example of 30 & 100 vertices and using penalization.	101
6.3	Graphs showing the best individual at each generation of the searching process for the algorithms UMDA, MIMIC, EBNA, cGA, eGA, and ssGA for the case of the 10 & 30 vertex graphs. Note the different scales between discrete EDAs and GAs. . . .	103
6.4	Graphs showing the best individual at each generation of the searching process for the algorithms UMDA, MIMIC, EBNA, cGA, eGA, and ssGA for the case of the 30 & 100 vertex graphs. Note the different scales between discrete EDAs and GAs. . . .	104
6.5	Graphs showing the best individual at each generation of the searching process for the algorithms UMDA, MIMIC, EBNA, cGA, eGA, and ssGA for the case of the 50 & 250 vertex graphs. Note the different scales between discrete EDAs and GAs. . . .	105
6.6	Graphs showing the best individual of the 10 & 30 case at each generation of the searching process for the continuous EDAs UMDA _c , MIMIC _c , EGNA _{BGe} , EGNA _{BIC} , EGNA _{ee} , and EMNA _{global}	109
6.7	Graphs showing the best individual of the 30 & 100 case at each generation of the searching process for the continuous EDAs UMDA _c , MIMIC _c , EGNA _{BGe} , EGNA _{ee} , and EMNA _{global}	109
6.8	Graphs showing the best individual of the 50 & 250 case at each generation of the searching process for UMDA _c , MIMIC _c , EGNA _{BGe} , and EGNA _{ee}	109
6.9	Figures showing the structures learned during the MIMIC search in discrete EDAs.	111
6.10	Figures showing the structures learned during the MIMIC _c continuous EDA.	112
6.11	Graphs showing the evolution of the Bayesian network in a EBNA search, illustrating clearly that the number of arcs of the probabilistic structure decreases gradually from the first generation to the last ones. A circular layout has been chosen in order to show the same nodes in the same position. The number of arcs decreases as follows respectively: 57, 56, 40, 12, 3, and 1. After the 37 th generation and until the last (the 43 th one) the Bayesian network does not contain any arc.	113

LIST OF FIGURES

6.12	Figures showing the structures learned during the Edge Exclusion EGNA _{ee} continuous EDA.	114
6.13	Figures showing the structures learned during the UMDA _c and EMNA _{global} continuous EDAs.	114
6.14	Graphs showing the evolution in number of arcs of the respective probabilistic structures for EBNA and EGNA _{ee} . The two different tendencies are illustrated: EBNA tends to a structure with less arcs when the search goes on, while EGNA-type algorithms tend to a structure with more arcs.	115
6.15	Illustration of the evolution in execution time when using MPI and depending on the number of processes.	120
6.16	Figures for the communication mechanisms and other MPI primitives on the parallel version of EBNA _{BIC} for the particular configuration of our cluster.	122
7.1	Illustration of the generation of the atlas (above) and the data graph (below). The atlas is created from a real 3D MR image of a healthy human brain manually segmented as we can see above. The data graph G_D is created segmenting the input image to be recognized. All the images are always in 3D, although here we show only a coronal view of them. Both graphs on the right show the complexity of the problem. The graph matching procedure has to assign a model vertex for each of the vertices in the data graph.	127
7.2	Example of the graph matching process applied to the recognition of structures in MR human brain images. In each row we have an axial and a sagittal view of the model graph, data graph, and a result respectively. All these images concentrate on the recognition of a particular segment of the brain among all the ones to be identified: the caudate nucleus.	128
7.3	Illustration of the process to generate the model graph: (a) an original image is selected to extract the model from; (b) a masked version which contains only the regions of interest around the landmarks is obtained; (c) the face model is obtained by manual segmentation; (d) image where the model is superimposed to the face image (just for explanatory purposes).	135
7.4	Example of an over-segmented image after applying the watershed algorithm.	136
7.5	Illustration of the first (reduced) example of the facial feature recognition problem. These images show the small parts selected from the images to create the model graph G_M (left) and the data graph (right).	138
7.6	Illustration of the solution obtained with the different EDAs for the first (reduced) example of the facial feature recognition problem. The regions encircled in white correspond to the segments matched as pupil and eyebrow satisfactorily, while the ones in red represent the errors in the solution.	140
7.7	Segmentation and recognition of facial features for the <i>deise</i> example using (a) eGA, (b) ssGA, (c) UMDA, (d) MIMIC and (e) EBNA _{BIC} . As the model has been extracted from this image, the target and the model image are the same for this case.	141
7.8	Segmentation and recognition of facial features for the example <i>f014</i> using (a) eGA, (b) ssGA, (c) UMDA, (d) MIMIC and (e) EBNA _{BIC}	142
7.9	Segmentation and recognition of facial features for example <i>f041</i> using (a) eGA, (b) ssGA, (b) UMDA, (c) MIMIC and (d) EBNA _{BIC}	143
7.10	Segmentation and recognition of facial features for the example <i>m036</i> using (a) ssGA, (b) UMDA, (c) MIMIC and (d) EBNA _{BIC}	144

7.11 Example of some typical error regions. A: two very small regions nearby facial features, which are too small to be properly identified. B: regions in the outer portion of the eyebrows that could be ambiguous to classify as eyebrow or skin, in this case recognized as skin. C: true matching error, in this case eyebrow region recognized as part of the pupil. 146

A.1 Example of three permutation-based individuals and a similarity measure $\varpi(i, j)$ between vertices of the data graph ($\forall i, j \in V_D$) for a data graph with $|V_D| = 10$. . . 154

A.2 Result of the generation of the individual after the completion of phase 1 for the example in Figure A.1 with G_D containing 6 vertices ($|V_M| = 6$). 154

A.3 Generation of the solutions for the example individuals in Figure A.1 after the first step of phase 2 ($|V_M| = 6$). 155

A.4 Result of the generation of the solutions after the completion of phase 2. 155

A.5 Example of redundancy in the permutation-based approach. The two individuals represent the same solution shown at the bottom of the figure. 156

B.1 Example of a Bayesian network structure. 158

C.1 Producer-consumer example with race conditions. 166

C.2 Example of the producer-consumer example solved with sleep & wake-up primitives. The solution proposed here is only valid for one producer and one consumer. 170

C.3 Example of solving the producer-consumer problem using semaphores. 171

C.4 Example of the producer-consumer scheme using message passing. 175

LIST OF FIGURES

List of Tables

5.1	Table showing the combination of communication models and parallel architecture models. The native communication models for multiprocessors and multicomputers are shared memory and message passing respectively.	89
5.2	Time to compute for two graph matching problems synthetically generated with sizes 10 & 30 (first column) and 50 & 250 (second column). All the figures are given in relative times, i.e. 100% = full execution time. The values with the symbol (*) would require more than a month of execution time to be properly computed.	93
5.3	Time to compute the analysis in Table 5.2 for the 10 & 30 and 50 & 250 examples (hh:mm:ss). Again, the values with the symbol (*) required more than a month of execution time to be properly computed.	94
6.1	Best fitness values for the 10 & 30 example (mean results of 20 runs).	106
6.2	Number of required generations for the 10 & 30 example (mean results of 20 runs).	106
6.3	Time to compute for the 10 & 30 example (mean results of 20 runs, in mm:ss format).	106
6.4	Best fitness values for the 30 & 100 example (mean results of 20 runs).	107
6.5	Time to compute for the 30 & 100 example (mean results of 20 runs, in hh:mm:ss format).	107
6.6	Best fitness values for the 50 & 250 example (mean results of 20 runs).	107
6.7	Time to compute for the 50 & 250 example (mean results of 20 runs, in hh:mm:ss format).	108
6.8	Particular non-parametric tests for the 10 & 30, 30 & 100 and 50 & 250 examples. The cases where the generations in GAs are $p = 1.000$ indicate that all GAs executed during 100 generations. These are the mean results of 20 runs for each algorithm.	108
6.9	Figures of the 3 cases of Study 1 for the continuous EDAs, obtained as the mean values after 20 executions of the continuous EDAs. The <i>best</i> column corresponds to the best fitness value obtained through the search.	110
6.10	Execution time for the 10 & 30 and 50 & 250 examples using the EBNA _{BIC} , EGNA _{BGe} and EGNA _{BIC} algorithms regarding their sequential and parallel versions of computing the BIC score (hh:mm:ss). The values with the symbol (*) required more than a month of execution time to be properly computed.	117
6.11	Execution times obtained when increasing the number of processes (processors used) for the medium-sized example with 30 & 100 vertices (above) and for the big example with 50 & 250 vertices (below). Times are presented in <i>hh:mm:ss</i> format.	121
7.1	Mean values of experimental results after 10 executions for each algorithm for the inexact graph matching problem of the human brain structure recognition.	130

7.2	Time of computing for the human brain recognition graph matching problem solved with EBNA _{BIC} . All the figures in the last column are given in relative times, i.e. 100 % = full execution time.	132
7.3	Execution times for the human brain recognition problem using the EBNA _{BIC} algorithm regarding its sequential and shared memory based parallel <i>pthread</i> version for computing the BIC score (hh:mm). Results show important improvements in execution times.	132
7.4	Execution times for the human brain recognition problem using the EBNA _{BIC} algorithm regarding its sequential and message passing based parallel MPI version for computing the BIC score (hh:mm). The execution time of the sequential version on one of the machines is 26 hours and 49 minutes. Results show the validity of the parallel system.	133
7.5	Summary of the results for the small facial feature recognition problem.	139
7.6	Figures of the 4 cases that are analyzed in the second example, illustrating the number of vertices and edges that are considered. These values are illustrated for showing the difference in complexity for each of the examples.	140
7.7	Figures of the 4 cases that we analyzed, illustrating the mean values after 5 executions of each of the algorithms. The <i>best</i> column corresponds to the mean best fitness value obtained through the search, and the differences between the algorithms are evident as EDAs obtain the best results for all the examples. The <i>time</i> column shows the CPU time required for the search, and the <i>eval.</i> one shows the number of individuals that had to be evaluated in order to end the search.	145
7.8	Statistical significance for all the 4 examples and algorithms after 5 executions of each of the algorithms, by means of the results of the non-parametric tests of Kruskal-Wallis and Mann-Whitney. The first column shows the result of the test comparing all EDAs with all GAs, the second is the test for comparing eGA and ssGA, and the third shows the comparison between the three EDAs.	145
7.9	Table showing the number of misclassified regions in each test image for each algorithm. The <i>Errors</i> column indicates the number of misclassified regions, while the column % shows the percentage with respect to the total number of regions in the image.	146