IDENTIFICACIÓN DE GALAXIAS USANDO ALGORITMOS DE APRENDIZAJE AUTOMATICO

ERICK RAMON MENESES CUADROS
LUIS GUILLERMO ORTIZ COVELLI

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICO MECÁNICAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
BUCARAMANGA
2006

IDENTIFICACIÓN DE GALAXIAS USANDO ALGORITMOS DE APRENDIZAJE AUTOMATICO

ERICK RAMON MENESES CUADROS
LUIS GUILLERMO ORTIZ COVELLI

Trabajo de Grado para Optar por el Título de
Ingeniero de Sistemas

Director:
Arturo Plata Gómez
Ph.D. en Ciencias para el Ingeniero.

Codirector:
Nelson Vera Villamizar
Ph.D. en Astrofísica

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICO MECÁNICAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
BUCARAMANGA
2006

A Dios por darme la oportunidad de vivir cada día,
brindarme lugar en el espacio
y un punto en la línea del tiempo.

A mi mamá, Edilia, por la educación
y su enorme esfuerzo
por brindarme todo en la vida.

A mi novia, Diana, por el amor,
complicidad y apoyo
en todas mis locuras.

A mi familia y amigos por estar
en los buenos y malos momentos.

ERICK RAMÓN MENESES CUADROS

A Dios, el supremo...

A mis padres Amelia y José Guillermo,
lo más importante en mi vida...

A mis hermanos Alcira y José Giovanni,
y claro, a Spot...

LUIS GUILLERMO ORTIZ COVELLI

AGRADECIMIENTOS

Los autores quieren expresar su agradecimiento a todas las personas y entidades que colaboraron en este proyecto de grado:

Al profesor Arturo Plata, por sus importantes aportes como director del proyecto.

Al profesor Nelson Vera, quien desde la distancia, en su calidad de codirector colaboró activamente en el proyecto.

Al ingeniero Víctor Martínez, por su valiosa colaboración desde el área del tratamiento digital de imágenes.

Al ingeniero Juan Carlos Escobar por su disposición.

Al profesor Henry Lamos por sus aportes desde el área de las matemáticas.

Al Grupo Halley, por ser un espacio de aprendizaje y avance de la ciencia.

A los profesores Fernando Ruiz y Alfonso Mendoza, por su constante apoyo.

A las ingenieras Diana Hortúa, Diana Rojo y Mónica Rivera, compañeras de batalla.

A nuestras familias, eje fundamental de nuestras vidas.

Y a todos nuestros amigos y amigas que estuvieron siempre a nuestro lado.

RESUMEN

Titulo: IDENTIFICACIÓN DE GALAXIAS USANDO ALGORITMOS DE APRENDIZAJE AUTOMATICO. [*]

Autores:
ERICK RAMON MENESES CUADROS [**]
LUIS GUILLERMO ORTIZ COVELLI [**]

Palabras Clave: Morfología Galáctica, Tratamiento Digital de Imágenes, Extracción de Parámetros, Aprendizaje Automático, Procesamiento Paralelo.

Descripción:
Este trabajo presenta un análisis experimental de algoritmos para la clasificación automática de imágenes de galaxias. La metodología usada se basa en tres etapas:

- El tratamiento digital de la imagen: Se estandarizan todas las muestras, cada imagen fue filtrada, rotada, centrada, ajustada a un tamaño estándar y dividida en sus componentes RGB. Se trabajó con la imagen final en varias formas: escala de grises, componentes RGB separados y componentes RGB unidos en una sola imagen.
- Extracción de parámetros: aquí se utilizo el Análisis de Componentes Principales como método eficaz para reducir la dimensionalidad de los datos y extraer la información relevante y propia de la imagen a identificar.
- Aprendizaje Automático: basados en los parámetros obtenidos en la fase anterior se usaron tres algoritmos: Redes Neuronales Artificiales, K-Vecinos más Cercanos y Regresión Localmente Ponderada, cuyo fin era analizar los datos y proveer un juicio acerca de la clase a la que pertenecía la imagen de la galaxia (espiral, elíptica o irregular).

Los mejores resultados obtenidos arrojaron un porcentaje del 85.89% de clasificación correcta para un conjunto de 429 imágenes usando el N-fold cross-validation como método de evaluación de los algoritmos.

Además, y de acuerdo a la mejor combinación de algoritmos obtenida se propone la mejora del método haciendo un diseño exhaustivo de su paralelización.

---

[*] Trabajo de Grado

[**] Facultad de Ingenierías Físico - Mecánicas
Escuela de Ingeniería de Sistemas e Informática

SUMMARY

Title: GALAXY IDENTIFICATION USING MACHINE LEARNING ALGORITHMS.[*]

Authors:
ERICK RAMON MENESES CUADROS[**]
LUIS GUILLERMO ORTIZ COVELLI[**]

Keywords: Galactic Morphology, Digital Image Processing, Parameter Extraction, Machine Learning, Parallel Processing.

Description: This work presents an experimental algorithm analysis for automatic galaxy images classification. The used methodology is based on three stages:

- Digital Image Processing: All samples are standardized. Each single image was filtered, rotated, centered, fixed to a standard size and divided in its RGB components. The final image was used in different ways: grayscale, separated RGB components and joined RGB components in a single image.

- Parameter Extraction: here, the Principal Components Analysis was used as an efficient method to reduce dimensionality of data and to extract relevant and unique information from the image.

- Machine Learning: based on the parameters obtained in previous stage, three algorithms was used: Artificial Neural Networks, K-Nearest Neighbors and Locally Weighted Regression, which goal was analyze the data and obtain a decision about galaxy image class (Spiral, Elliptical or Irregular).

Best experimental results obtained, gives 85.89% of correct classification for a set of 430 images, using the N-Fold cross validation as a method to evaluate those algorithms.

Additionally, according to the best algorithm combination obtained, the improvement of the method was proposed, making a complete design of its parallelization.

---

[*] Grade Work

[**] Faculty of Physics – Mechanics Engineering
Engineering of Systems and Informatics

CONTENIDO

LISTA DE FIGURAS

pág.

LISTA DE TABLAS

pág.

1. INTRODUCCION

La Astronomía es la ciencia más antigua estudiada por el hombre, y al mismo tiempo es de la que más información se tiene, pues el desarrollo de herramientas tecnológicas necesarias para un buen conocimiento en esta área, nos ha permitido desde hace casi 400 años, desde la invención del telescopio, observar el entorno en donde nos encontramos (a gran escala) y poder entender situaciones y fenómenos de los que antes se carecía de información.

Gracias a los grandes telescopios, hoy en día contamos con una gran cantidad de datos y de información acerca de las galaxias, en forma de imágenes, ondas de radio, espectros, etc. La mayor parte de la información espera por ser analizada y divulgada,
y cada día es mayor el flujo de datos y de información obtenida.

La morfología de galaxias estudia específicamente la composición física de la galaxia, caracterizándola e individualizándola para clasificarlas en sus distintas clases.
Aún cuando el esquema de Morfología / Clasificación se ha convertido en un gran conjunto complejo de designaciones, las características básicas observables son simples: tamaño relativo y el brillo del núcleo con respecto al disco o la cubierta exterior (cuantificada como la relación bulbo – disco), la forma del disco (Ej. brazos, anillos, barras o asimetrías), y otros factores tales como el brillo de la superficie, color y la presencia de polvo.

Específicamente, en cuanto a las imágenes de galaxias, se puede decir que algunos *surveys* (Proveedores de imágenes) tienen alrededor 10^8 fotografías de galaxias, y con las constantes mejoras en el campo de la detección (óptica, radioastronomía, etc.) este número va en aumento.

La clasificación de estas imágenes se realiza usualmente mediante inspección visual o mediante placas fotográficas. Sin embargo, esta tarea no es nada fácil, porque se requiere bastante experiencia y habilidad. También se consume mucho tiempo: catálogos que contienen clasificaciones humanas, llevan años en completarse y contienen sólo decenas de miles de registros.

Esto necesariamente implica que se tenga que recurrir a procesos de cómputo muy fuertes para analizar y clasificar estas imágenes de una manera práctica.

Este proyecto pretende aportar desde la ingeniería de sistemas, una posible solución al problema presentado anteriormente. Para ello nos hemos planteado el siguiente objetivo:

> Implementar un algoritmo que, a partir de la imagen apropiada de una galaxia y mediante un proceso de tratamiento de imágenes, obtenga unos parámetros definidos, que permitan decidir mediante el uso de técnicas de aprendizaje automático la clase a la que pertenece.

Además, como mejora al proceso secuencial, presentamos el diseño paralelo del mismo, con la finalidad de obtener beneficios computacionales como ahorros en los tiempos de cómputo y aumento en la eficiencia del algoritmo.

Para lograr estos objetivos, se pretende aplicar diferentes técnicas de tratamiento de imágenes y algoritmos de aprendizaje automático.
Se espera que los resultados del proyecto, sirvan para sentar las bases de una investigación más profunda en el área de clasificación morfológica de galaxias, de la cual no hay antecedentes en nuestro país.

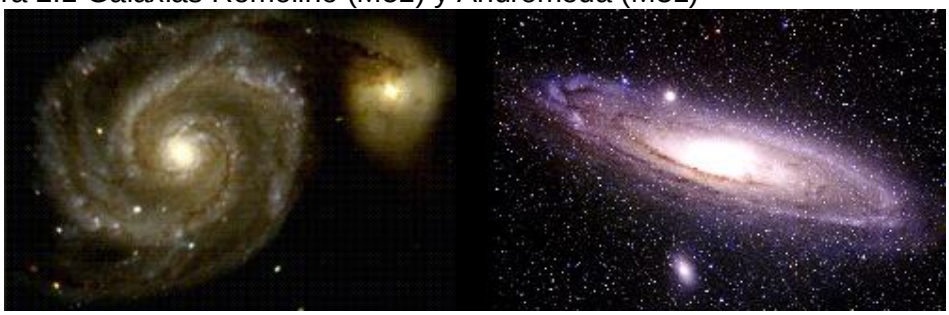
2. FUNDAMENTACION TEORICA

2.1 MORFOLOGÍA GALÁCTICA

Las galaxias son agrupaciones a gran escala de estrellas (del orden de cientos de miles de millones de estrellas en una galaxia). Nuestro sistema solar, al cual pertenece la Tierra, hace parte de la galaxia La Vía Láctea.

Al contrario de las estrellas, las galaxias observadas a través de un telescopio suficientemente potente, lucen como nubes difusas sin forma definida, pero si se captura la imagen de una galaxia en una placa fotográfica con un tiempo de exposición grande (de 30 minutos a dos horas), aparecen entonces detalles claramente definidos de su estructura morfológica, proporcionando así características que permiten luego definir la clase a la cual pertenece la galaxia observada.

Figura 2.1 Galaxias Remolino (M51) y Andrómeda (M31)



fuente: http://antwrp.gsfc.nasa.gov

A principios del siglo XX, Edwin Hubble se dedicó a observar objetos que hasta ese momento se denominaban "nebulosa", pues su aspecto era similar al de una nube. Hubble descubrió que estos objetos no pertenecían a nuestra galaxia y que además eran otras galaxias. También se dedicó a medir sus distancias y clasificarlas.

La ciencia de la Morfología de galaxias se concretó con el trabajo de Hubble (1926 y 1936), cuya famosa “secuencia de Hubble” continúa siendo una técnica estándar para el estudio de galaxias. Esta secuencia de galaxias espirales normales y elípticas fue expandida y delineada en subclases por de Vaucouleurs en su documento clásico en 1959 también descrito en el *Atlas de Hubble* (Sandage 1961 y 1975).

Más recientes revisiones se encuentran en Buta (1992), Roberts y Haynes (1994) y Van Den Bergh (1998). Aún cuando el esquema de Morfología / Clasificación se ha convertido en un gran conjunto complejo de designaciones, las características básicas observables son simples: tamaño relativo y el brillo del núcleo con respecto al disco o la cubierta exterior (cuantificada como la relación bulbo – disco), la forma del disco (ej. brazos, anillos, barras o

asimetrías), y otros factores tales como el brillo de la superficie, color y la presencia de polvo.

La secuencia de Hubble, a primera vista, parece ser simplemente una "guía" cualitativa que da los nombres convenientes a la amplia variedad de galaxias observadas en el Universo. Pero como señala Sandage (1986), la secuencia de hecho separa las galaxias en clases físicamente relacionadas, representando una secuencia "evolutiva". Por ejemplo, la edad promedio del disco de las espirales progresa a lo largo de la secuencia (de los primeros a los últimos tipos), así como la tasa actual de la formación de estrellas (ej, las galaxias de último tipo están formando estrellas a una tasa muy superior que las galaxias del tipo primero, Ferrini y Galli, 1988). Hubble y Humason (1931) entendieron que la secuencia morfológica estaba correlacionada con el ambiente de densidad de la galaxia: los cúmulos de galaxias contienen más galaxias lenticulares y elípticas que la densidad de campo típica.

La segregación morfológica parece ser una propiedad de los grandes cúmulos, incluyendo los de Virgo y Coma. Más allá, la función de luminosidad de las espirales es diferente de la de las galaxias de tipo elíptico (Binggeli y otros 1988). Aunque el origen y la formación de galaxias elípticas aun no se conoce plenamente, trabajos más recientes, han apuntado a un escenario en el cual las galaxias elípticas podrían ser el resultado de fusiones de galaxias espirales en los densos núcleos de cúmulos de galaxias (ejm. Dressler 1980, Postman y Geller 1984, Mamon 1992, Whitmore y otros 1993, Julian y otros 1997). Hay evidencia que los tipos morfológicos muy separados en la Sucesión de Hubble, tienen colores principales sistemáticamente diferentes, presumiblemente de diferentes poblaciones de estrellas, formadas antiguamente y recientemente, que dominan la luz (ejm. Holmberg 1958, de Vaucouleurs 1977, Giovanelli y Haynes 1983, Roberts y Haynes 1994, Buta y otros 1994, Odewahn y otros. 1996). Otra clase de galaxia, la enana esferoidal, también parece asociarse con los ambientes densos de los cúmulos, aunque su origen, evolución, edad y existencia sostenida (de las fuerzas de marea gravitatorias disociadoras), aún es un misterio (Sandage 1990, Impey y Bothun 1997).

Galaxias Elípticas

Tienen forma de elipse, como su propio nombre lo indica. Están compuestas en su totalidad por estrellas viejas, tan viejas como la propia galaxia; esta es la causa del color rojizo que las caracteriza.

El tamaño de las Galaxias Elípticas varía desde enanas (1 millón de estrellas) de 1000 años luz de diámetro (un año luz equivale a la distancia recorrida por la luz en 1 año, a 300000 km. por segundo, aproximadamente 9.47x10^12 km.), hasta galaxias muy masivas (1 billón de estrellas) de 100000 años luz o más.

Figura 2.2 Galaxia Elíptica M110

fuente: http://content.answers.com/

Galaxias Espirales

Tienen un núcleo más prominente que el de las demás galaxias, alrededor del cual giran los brazos. El conjunto de brazos se denomina disco, el cual posee, además de estrellas, nebulosas y bandas oscuras de polvo.

El diámetro de las Galaxias Espirales varía entre 15000 y 150000 años luz y contienen entre 10000 millones y 10 billones de estrellas; además giran en torno a su núcleo a una velocidad de alrededor de 300 km/s.

Las Galaxias Espirales representan un 75 % del total de las galaxias. La galaxia a la cual pertenece el sistema solar (en el que se encuentra la Tierra), es una galaxia Espiral y se llama La Vía Láctea, la Galaxia de Andrómeda es la más cercana a la Vía Láctea y también es Espiral.

Figura 2.3 Galaxia Espiral NGC1232

fuente: http://antwrp.gsfc.nasa.gov

Galaxias Espirales Barradas

Son mucho menos frecuentes que las normales. En ellas, las estrellas más brillantes y el gas caliente de las regiones centrales se organizan en una barra

recta que se extiende varios miles de años luz a ambos lados del centro, antes de curvarse alrededor de la galaxia para formar los brazos en espiral.

Figura 2.4 Galaxia Espiral Barrada NGC1300

fuente: http://antwrp.gsfc.nasa.gov

Galaxias Lenticulares

Las Galaxias Lenticulares tienen una morfología intermedia entre las Elípticas y las Espirales. Poseen un bulbo central y un disco transversal. Debido a su color rojizo, se puede asegurar que la componen estrellas en edad avanzada, como ocurre en las Elípticas.

Figura 2.5 Galaxia Lenticular Sombrero (M104)

fuente: www.unet.univie.ac.at

Galaxias Irregulares

Tienen un aspecto asimétrico, y las estrellas calientes no forman ramas espirales, sino que se concentran en grupos aislados o están dispersas por todo el disco galáctico. Contienen estrellas viejas y jóvenes con una gran cantidad de gas y polvo (hasta un 30 % de la masa de estas galaxias puede estar en forma de gas). Un ejemplo de galaxias Irregulares son la Gran y Pequeña nube de Magallanes, galaxias satélites de la Vía Láctea.

Figura 2.6 Galaxia Irregular Gran Nube de Magallanes (LMC)

fuente: www.astrocosmo.cl

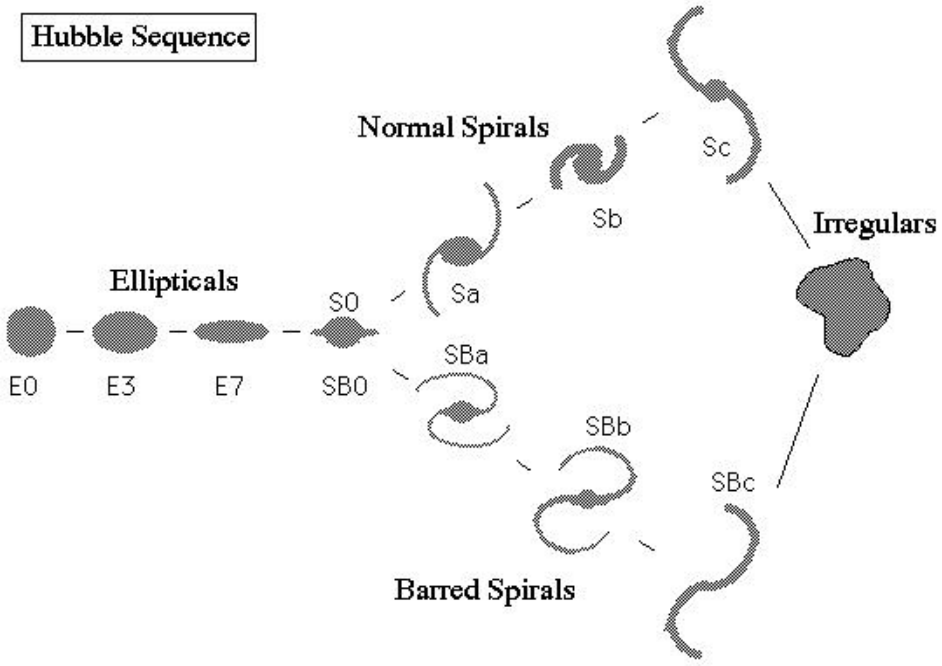
La secuencia de Hubble

Edwin Hubble realizó una valiosísima contribución a la astronomía, al desarrollar el primer esquema de clasificación, basado en la apariencia visual de las galaxias. Este esquema se conoce como La Secuencia de Hubble y en él se distinguen varios tipos morfológicos de galaxias, los ya mencionados Elíptica, Espiral e Irregular.

Las Galaxias Elípticas se subdividen de acuerdo al grado de elipticidad, que es la medida de achatamiento y viene dada por $e = 10(a-b)/a$, siendo a y b los semiejes mayor y menor de la elipse respectivamente. Una galaxia de aspecto circular ($e = 0$) se clasifica como E0 y la de aspecto más elíptico ($e = 7$) se denomina E7.

Las galaxias espirales, tanto las normales (S) como las barradas (SB) van desde los tipos tempranos (Sa, SBa) hasta tipos tardíos o evolucionados (Sc, SBc). Los tipos tempranos tienen brazos muy cerrados y con poco contraste mientras que el bulbo (o núcleo) es grande. La galaxias Lenticulares se clasifican como S0 y SB0.

Figura 2.7 Secuencia de Hubble



2.2 TRATAMIENTO DIGITAL DE IMÁGENES

Fundamentos.
El tratamiento de imágenes surge como un área de la computación con el fin de resolver problemas al hacer gigantescos envíos de información gráfica, junto a las dificultades que presentaban las agencias aeroespaciales con la información que recibían de las sondas espaciales y satélites artificiales que por aquel entonces empezaban a revelar las maravillas de nuestro universo respecto a la topografía y condiciones ambientales de los planetas y muchos otros objetos mas allá de el limite conocido por el hombre hasta ese momento.

Se puede decir que el mayor desarrollo en el área del tratamiento digital de imágenes se da en la década de los ochenta cuando se incrementó la construcción de equipos con mayor capacidad de cómputo e implementación de modelos matemáticos que permitían extraer las características de una imagen, además se desarrollaron avances significativos en otras áreas del conocimiento y la industria.

Básicamente el tratamiento de imágenes se ocupa de las siguientes etapas: Adquisición, preprocesamiento, segmentación, descripción y reconocimiento.

Adquisición.
Mediante un dispositivo sensible a una determinada banda del espectro electromagnético (rayos X, ultravioleta, visible o infrarrojo) se produce una señal eléctrica de salida proporcional al nivel de energía detectado y mediante otro dispositivo llamado digitalizador se convierte esta señal en un formato digital, obteniéndose de esta manera la imagen.

El dispositivo convertidor análogo digital debe tener en cuenta dos criterios muy importantes al momento de realizar la discretización de las intensidades: e*l muestreo*, que relaciona la dimensión espacial e indica el tamaño del elemento de la imagen (píxel), el segundo criterio hace referencia a la amplitud de la señal, es decir, el nivel de intensidad del píxel, conocido como *la cuantificación*.

Figura 2.8 Galaxia Espiral vista de canto

fuente: http://antwrp.gsfc.nasa.gov

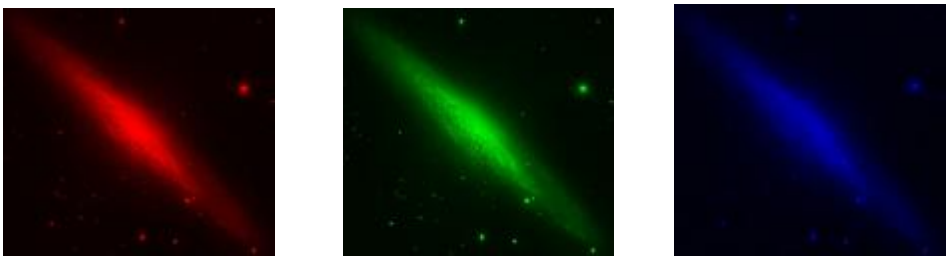
Para el caso de la astronomía, se ha generalizado el uso de una cámara acoplada al telescopio que tiene un dispositivo de cargas eléctricas interconectadas, conocida como CCD (del inglés Charge-Coupled Device), el cual proporciona una sensibilidad en la captura de la imagen considerablemente superior al de las cámaras normales.

Preprocesamiento.
En el proceso de adquisición, la imagen que se genera no es totalmente confiable, dado que se pueden presentar defectos como fallos en el delicado sistema óptico del telescopio, partículas de polvo en el objetivo (lente primario del telescopio), aberraciones crómica y esférica, etc.

El filtrado de una imagen permite corregir algunos de estos problemas y ha sido agrupado conforme a la técnica utilizada. Cada filtro es implementado de acuerdo a las características de la imagen que se desea obtener.

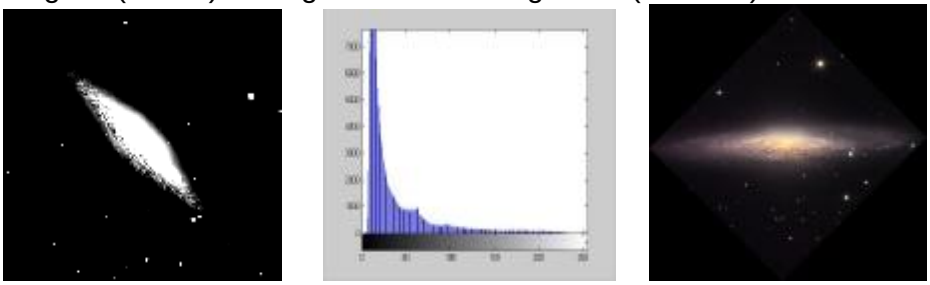
- Descomposición en bandas. Las imágenes a color, en realidad son una superposición de las imágenes tomadas a diferentes longitudes de onda (dentro del espectro electromagnético).
  Al descomponer la imagen, cada banda representa una región diferente y por lo tanto se tiene información sobre el comportamiento del objeto (galaxia) en esa parte del espectro (altas energías, polvo estelar, etc.).

Figura 2.9. Bandas R (izquierda), G (centro) y B (Derecha) de la imagen de la galaxia



- Mejoramiento por procesamiento de punto. Se basa en la mejora de las intensidades de los niveles de grises considerando los píxeles individualmente. Entre los más utilizados están: Negativos de imágenes, Aumento del Contraste, Compresión del Rango Dinámico, Procesamiento del Histograma (Ecualización y Especificación) y Transformaciones de Rotación.

Figura 2.10. Imagen binarizada de la galaxia (izquierda), histograma de la imagen original (centro) e imagen rotada de la galaxia (derecha).

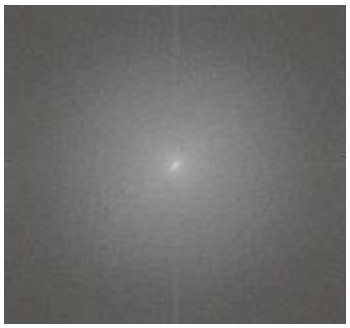


- Filtrado en el Dominio Espacial. El filtrado espacial de una imagen es implementado mediante pequeñas matrices cuadradas, conocidas como máscaras, núcleos o ventanas, las cuales, aplicando el teorema de la convolución con la imagen a filtrar, produce como salida otra imagen donde ha sido modificado el contenido en escala de grises. A su vez han sido clasificados como Paso-Bajo y Paso-Alto, los primeros permiten homogenizar las superficies de la imagen mediante el difuminado, los segundos son útiles en la extracción de bordes. Entre los filtros espaciales más comunes encontramos: El Filtro Mediana, El Filtro Promedio y los Filtros Diferenciales (Operadores de Roberts, Prewitt y Sobel).

Figura 2.11. Imagen de la galaxia en dominio espacial

- Filtrado en el Dominio de la Frecuencia. Para implementar esta serie de filtros se calcula la Transformada de Fourier de una imagen, con el fin de obtener las componentes frecuenciales de la imagen para multiplicarlas por una función de transferencia de un filtro, seguidamente se aplica la transformada Inversa de Fourier para producir la imagen filtrada. Estos se encuentran clasificados como Paso-Bajo y- Paso-Alto, los primeros atenúan las componentes de alta frecuencia y los segundos las bajas.
  La función de transferencia más utilizada es conocida como Filtro de Butterworth tanto para frecuencias altas como para frecuencias bajas.

Figura 2.12. Representación de la imagen de la galaxia en espacio frecuencial



Segmentación.
Esta etapa nos permite descomponer en sus partes constituyentes los objetos que se encuentran en la imagen (objetos extraños: estrellas, rayos cósmicos, etc.. y la galaxia como tal: núcleo galaxia, brazos, etc.).
De esta etapa depende el éxito o fracaso del análisis de la imagen. Se utilizan diversos algoritmos con el fin de detectar discontinuidades, líneas y bordes. Entre estos algoritmos se encuentran el Laplaciano de una función

bidimensional, la Transformada de Hough y el procesado global por medio de teoría de grafos.

Figura 2.13. Imagen centrada y recortada de la galaxia

En la segmentación de la imagen se utiliza también la técnica de umbralización, la cual consiste en convertir una imagen en escala de grises a una imagen binaria (Blanco y Negro puros) utilizando uno de los niveles de gris como umbral. Se acepta que los objetos son aquellas regiones que dan como resultado el color blanco mientras que la región negra pertenece al fondo de la imagen.

Descripción.
Una vez realizada la segmentación de la imagen se continúa a la etapa de descripción, consistente en describir las características de los objetos presentes en la imagen respecto a sus características externas (contorno) y sus características internas (píxeles que comprenden la región). Esta etapa permite diferenciar cada una de las características físicas de la galaxia (núcleo, numero de brazos, elipticidad, simetría, etc.). Entre los esquemas de representación para contornos encontramos:

- Aproximaciones Poligonales. El objetivo es lograr una aproximación poligonal de un contorno con el menor número de lados posibles.

- Descriptores Geométricos. Se utilizan medidas geométricas como; longitud, diámetro, área, centroide, etc.

- Descriptores de Fourier. Son los coeficientes de la Transformada Discreta de Fourier de una serie, donde las coordenadas de los píxeles son utilizados como una serie de números complejos.

- Descriptores Matemáticos. Son los valores propios y vectores propios que identifican el objeto analizado y resultan de aplicar el método de Análisis de Componentes Principales.

Reconocimiento.

En esta etapa se asigna una etiqueta a los objetos de acuerdo con sus descriptores permitiendo reconocer e identificar patrones de los objetos para ser comparados con otros patrones y poder así reconocer e identificar el objeto de la imagen que se ha analizado. Mediante algoritmos como los de mínima distancia y los de aprendizaje por ejemplos, el proceso que se la ha aplicado a la imagen permite establecer criterios de decisión.

2.3 ALGORTIMOS DE EXTRACCIÓN DE PARÁMETROS

La extracción de parámetros es un proceso común, lo hacemos a diario pero no nos damos cuenta de ello, para la mayoría de las cosas tenemos grupos o clases, por ejemplo: cuando hablamos de las personas las clasificamos de muchas maneras: por estrato social, por región, nivel intelectual, etc... y el que pertenezcan a una clase u otra lo sabemos de acuerdo a unos parámetros que inconscientemente extraemos y evaluamos comparándolos con unos valores base.

Esos parámetros pueden ser de diversas índoles, por ejemplo: para decir que una persona es de la costa pacífica tal vez miraríamos su color de piel, cabello, ancho de la nariz y otros parámetros físicos que si se encuentran dentro de un rango cerca al promedio podríamos concluir rápidamente que nuestra afirmación es cierta. Por otro lado para decir que una persona es inteligente trataríamos de medir el nivel de conocimiento, la agilidad mental, capacidad de memorizar y otros parámetros que ya no se encuentran dentro de un espacio físico.

Dentro de este trabajo se busca caracterizar las diferentes clases de galaxias y para ello tenemos que obtener parámetros que brinden una buena información y ayuden a identificar el grupo al que pertenece.

A continuación se muestran algunas formas (las mas conocidas) de hacer extracción de parámetros para la clasificación de imágenes Galaxias, todas ellas fueron estudiadas y se muestra una definición básica de cada una, sin embargo se hace énfasis en la técnica que se escogió para el desarrollo del trabajo (análisis de componentes principales).

Descripción Usando Parámetros Fotométricos

Una de las formas mas comunes de hacer descripción morfológica da galaxias es a través de parámetros fotométricos, es decir, parámetros físicos propios de la galaxia extraídos de la imagen mediante un adecuado tratamiento digital.

Entre los parámetros mas comunes encontramos la concentración de luz hacia el centro de la galaxia (Morgan 1958), la concentración usando fotometría isophotal (Doi 1993), la simetría rotacional y flujo en aperturas elípticas (Abraham 1996), la energía para altas frecuencias espaciales (Takamiya 1999) y el índice de textura (Yamauchi 2005) entre muchos otros.

Mediante la extracción de un buen grupo de parámetros y una asignación de pesos según su relevancia se puede llegar a una buena clasificación, de hecho la mayoría de trabajos para clasificación de galaxias se han dado en esta

forma, sobre todo los más antiguos, aunque aun continua siendo un buen método de caracterización.

Descripción de Imágenes Astronómicas Aplicando Shapelets
Este método hace una descomposición lineal de cada objeto sobre la imagen en una serie de funciones base localizadas, para las diferentes formas que se presentan, dichas funciones son denominadas “Shapelets”.

Shapelets es un sistema completo, dado por un conjunto ortonormal de funciones base 2D construidas a partir de los polinomios de Laguerre o Hermite que corresponden a las perturbaciones alrededor de una función Gaussiana circular, donde una combinación linear de estas funciones se puede utilizar para modelar cualquier imagen, de una manera similar a la síntesis de Fourier o Wavelet.

La descomposición del shapelet es particularmente eficiente para las imágenes localizadas en espacio, y proporciona un de alto nivel de la compresión para las galaxias individuales en datos astronómicos, además tiene muchas características matemáticas elegantes que la hacen conveniente para el análisis y el procesamiento de imagen.

Descripción a partir de Espacios de Frecuencias
Es sabido que para ciertas operaciones sobre una imagen, es muy útil pasar a un espacio de frecuencias donde se hacen mas evidentes ciertas características, y estando allí, podemos hacer filtrados, aplicar transformadas, hacer descripción, etc... optimizando los resultados.

Con el uso del espacio de frecuencias para estudio morfológico de galaxias se pueden aplicar variaciones de la transformada de Fourier sobre la imagen para encontrar la velocidad del patrón espiral y medir características morfológicas como el pitch angle (ángulo para los brazos de la galaxia) y la distribución radial de densidad (Vera 2004), en el caso de las galaxias espirales.

Descripción a través de Análisis de Componentes Principales
El objetivo principal que persigue el ACP es la representación de las medidas numéricas de varias variables en un espacio de pocas dimensiones donde nuestros sentidos puedan percibir relaciones que de otra manera permanecerían ocultas en dimensiones superiores. Dicha representación debe ser tal que al desechar dimensiones superiores la pérdida de información sea mínima.

Un símil podría ilustrar la idea: imaginemos una gran lámina rectangular (objeto de tres dimensiones) de por ejemplo, 3m de larga, 2m de ancha y 4 cm de

espesor. Para efectos prácticos, dicha lámina puede ser considerara como un objeto plano (de dos dimensiones) de 3m de largo por 2m de ancho.
Al realizar esta reducción de dimensionalidad se pierde cierta cantidad de información ya que, por ejemplo, puntos opuestos situados en las dos caras de la lámina aparecerán confundidos en un solo. Se pierden las distancias perpendiculares a las caras. Sin embargo, la pérdida de información se ve ampliamente compensada con la simplificación realizada, ya que muchas relaciones, como la vecindad entre puntos, es más evidente cuando éstos se dibujan sobre un plano que cuando se hace mediante una figura tridimensional que necesariamente debe ser dibujada en perspectiva.

Por otra parte, las aplicaciones del ACP son numerosas, entre ellas podemos citar la clasificación de individuos, comparación poblacional, estratificación multivariada, etc.

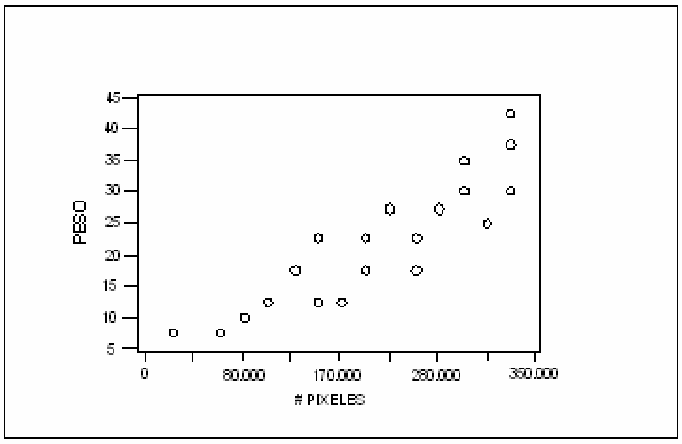
Tomemos un ejemplo sencillo a fin de entender mejor el funcionamiento: supóngase que se mide el peso en kilo bites y el número de píxeles de 10 imágenes y que se obtienen los siguientes datos:

Tabla 2.1 Datos para el ejemplo de las imágenes

IMAGEN	PESO	# PIXELES		IMAGEN	PESO	# PIXELES
1	2	16.129		6	21	169.350
2	4	33.870		7	26	150.210
3	8	67.500		8	33	270.000
4	12	74.514		9	40	332.500
5	18	104.300		10	45	345.560

Los datos anteriores pueden ser dibujados mediante un diagrama de dispersión así:

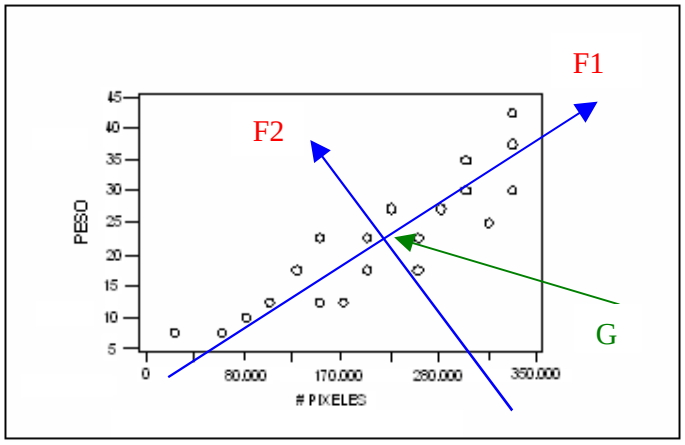
Figura 2.14 Diagrama de dispersión del ejemplo de las imágenes



Como se puede ver, cada variable puede representarse sobre un eje coordenado y así cada pareja de valores (x_i, y_i) representa las medidas del i-ésimo individuo, los cuales al ser representados en el plano forman la nube de individuos.

Ahora, se quiere construir un nuevo sistema de coordenadas ortogonales en el cual los puntos puedan ser representados de una manera tal que sus proyecciones sobre el nuevo primer eje recojan la mayor cantidad posible de variación y las proyecciones sobre el segundo eje recojan el resto de variación. Intuitivamente se puede encontrar que tales ejes corresponden a las rectas F1 y F2, representadas en la siguiente gráfica cuyo origen se encuentra en el centro de gravedad G de la nube de puntos (cuyas coordenadas son las medias de las variables), tal como se ve en la figura:

Figura 2.15. Nuevo sistema de coordenadas para el ejemplo de las imágenes



El nuevo sistema de coordenadas con origen en el centro de gravedad de la nube de puntos cuenta con dos ejes: el primero (F1) que recoge la mayor cantidad posible de variación y el segundo (F2) que tiene la cantidad de variación restante, además podemos observar claramente dos movimientos: uno de translación que sitúa el nuevo origen en el centro de gravedad de la nube y uno *rotación*, usando el centro de gravedad como punto pivote.

Es evidente que el nuevo sistema de coordenadas tiene entonces tantos ejes perpendiculares entre sí como tenía el antiguo, es decir, tantos ejes como variables se hayan considerado inicialmente.

Ahora, desarrollemos el proceso matemático que involucra el análisis de componentes principales, tomemos *p* variables aleatorias de tipo numérico $X_1, X_2, \mathbf{L}, X_p$ (las cuales posiblemente estén correlacionadas entre sí), entonces:

- Primero consideremos las variables $X_1, X_2, \mathbf{L}, X_p$ conjuntamente para formar un vector denotado por: $\mathbf{X} = (X_1, X_2, \mathbf{L}, X_p)$.

$$\begin{array}{ll} \boldsymbol{X_1} = & \begin{pmatrix} x_1 & x_2 & x_3 & ----- & x_n \end{pmatrix} \\ \boldsymbol{X_2} = & \begin{pmatrix} x_1 & x_2 & x_3 & ----- & x_n \end{pmatrix} \\ \mathbf{.} & \begin{pmatrix} . & . & . & & . \end{pmatrix} \\ \mathbf{.} & \begin{pmatrix} . & . & . & & . \end{pmatrix} \\ \boldsymbol{X_p} = & \begin{pmatrix} x_1 & x_2 & x_3 & ----- & x_n \end{pmatrix} \end{array}$$

- Ahora a partir de esa variable **X** vamos a construir su matriz de varianzas covarianzas definida como M y donde en la fila i columna j tiene el valor de la covarianza entre X_i y X_j, por lo que su diagonal esta conformada por las varianzas $\mathrm{V}(X_1), \mathrm{V}(X_2), \mathbf{L}, \mathrm{V}(X_p)$, esta matriz es simétrica y diagonalizable.
  Para eso calculamos la media y la desviación estándar para cada variable X_a:

$$\bar{x}_a = \frac{1}{n} \cdot \sum_{i=1}^{n} x_{ai} \qquad \text{y} \qquad S_a = \sqrt{\frac{1}{n-1} \cdot \sum_{i=1}^{n} (x_{ai} - \bar{x}_a)^2}$$

  Con esos datos, ya podemos estandarizar las distintas variables (transformar un conjunto de datos en otro, con media cero y desviación estándar uno). Pasamos de la variable X_a a la Z_a de esta forma:

$$z_{ai} = \frac{x_{ai} - \bar{x}_a}{s_a} \qquad Z = \begin{pmatrix} z_{11} & z_{12} & \cdots & z_{1n \cdot m} \\ z_{21} & z_{22} & \cdots & z_{2n \cdot m} \\ \vdots & \vdots & \ddots & \vdots \\ z_{p1} & z_{p2} & \cdots & z_{pn \cdot m} \end{pmatrix}$$

Ahora, teniendo ordenadas las variables estandarizadas puedo calcular la matriz de covarianza mediante la siguiente operación matricial:

$$M = \frac{1}{n} \cdot Z \cdot Z^t$$

- El siguiente paso es calcular los valores y vectores propios de la matriz de correlación calculada. Los valores propios son las raíces del polinomio:

$$\det(M - \lambda I) = 0$$

Donde I representa la matriz identidad, de las mismas dimensiones que la matriz M. Esta expresión da como resultado un polinomio cuyas raíces serán los valores propios de M, que se denotan como $l_1, l_2, \mathbf{L}, l_p$

Los vectores propios V_1, V_2, ...V_p asociados a esos valores propios, se calcularán sustituyendo los valores propios en la fórmula:

$$(M - \lambda I) \cdot v_i = 0$$

Para cada valor propio λ_i, obtenemos una ecuación diferente, y de esta ecuación obtenemos también un vector propio v_i diferente y asociado a su respectivo λ_i.

- Las coordenadas de los vectores propios hallados son los coeficientes de la transformación que hay que realizar para pasar de las variables originales a las nuevas variables 'componentes principales'.

Los valores propios nos dan el orden en el que hay que poner esos vectores propios; el valor propio mayor nos está indicando que su vector propio asociado apunta en la dirección de máxima variabilidad de los datos, es decir, en la de la primera componente principal; el segundo valor propio hace lo mismo con su vector propio, indicando que apunta en la siguiente dirección de máxima variabilidad ortogonal con la anterior, y así sucesivamente. Es por ello que la obtención de los componentes principales se realiza de la forma:

$$CP = Z^t \cdot V$$

donde Z es la matriz de valores estandarizados, aunque también se podría emplear X (la de valores originales), y V es una matriz de p filas y q columnas, que recoge todos los vectores propios, ordenados según valores propios. Podemos desarrollar uno de los elementos de la matriz CP de componentes principales:

$$CP_{ij} = \sum_{k=1}^{p} v_{kj} \cdot z_{ki}$$

Obteniendo las variables CP_1, CP_2, ..., CP_p. Podemos realizar simples cálculos para comprobar que:

$$\overline{CP_i} = 0 \quad \text{(la media es cero para todas)}$$
$$S^2_{CP_i} = \lambda_i \quad \text{(la varianza es el valor propio)}$$
$$S_{CP_i CP_j} = 0 \quad \text{(están totalmente decorrelad as)}$$

El proceso anteriormente descrito genera nuevas variables (componentes), mediante una combinación lineal de las p variables originales, pero donde los vectores o componentes generados tienen unas características especiales.

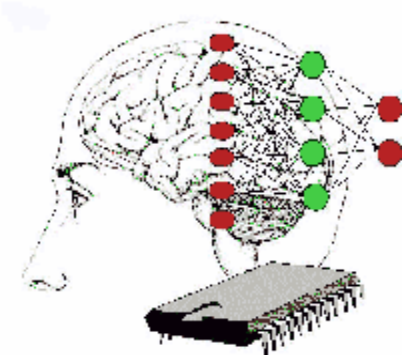
Para nuestro trabajo es importante tener en cuenta que ese grupo de variables originales pertenece a un grupo de individuos que en nuestro caso son las diversas imágenes a clasificar. La aplicación del ACP a imágenes:

- Busca construir una o varias imágenes que incrementen su capacidad de diferenciar distintas coberturas. Es por ello que al realizar una composición color resulta interesante usar, en lugar de algunas bandas de la imagen, los componentes principales 1, 2 y 3 en la secuencia RGB respectivamente.
- Puede aplicarse como realce previo a la interpretación visual o como procesamiento anterior a la clasificación. En general, esta técnica incrementa la eficiencia computacional de la clasificación porque reduce la dimensionalidad de los datos.
- Facilita una primera interpretación sobre los ejes de variabilidad de la imagen, lo que permite identificar aquellos rasgos que aparecen en la mayoría de las bandas y aquellos otros que son específicos de algún grupo de ellas.

2.4 ALGORITMOS DE APRENDIZAJE AUTOMATICO

El aprendizaje automático es la rama de la inteligencia artificial que estudia como construir sistemas computacionales que, a través de la experiencia, mejoren su desempeño. En otras palabras, a medida que se les suministre ejemplos, van generalizando un comportamiento de salida (respuesta) de modo que ante una entrada nueva responda con una salida adecuada.
Como el proceso en general trata de imitar al cerebro humano, se puede decir que el aprendizaje automático es un proceso de inducción del conocimiento.

Figura 2.16 Aprendizaje Automático emulando el cerebro humano



fuente: HILERA GONZALEZ, José Ramón. Redes Neuronales Artificiales: Fundamentos, Modelos y Aplicaciones.

Taxonomía
La taxonomía de los algoritmos de aprendizaje automático se puede establecer, según el tipo de problema a resolver, en algoritmos de aprendizaje supervisado, aprendizaje no supervisado y aprendizaje por refuerzo; según el tipo de la representación de la información o de los datos en preposicional o atributo–valor y lógica de primer orden o relacional; y según el tipo de proceso de aprendizaje en inductivo o deductivo.

Según el tipo de problema a resolver

- Algoritmos de aprendizaje supervisado:
  En este tipo de aprendizaje, el sistema *aprende* mediante una función que establece una correspondencia entre las entradas y las salidas, es decir, para cada dato se proporciona una salida deseada. El sistema se alimenta mediante ejemplos previamente etiquetados (se conocen la entrada y la salida correspondiente).

Algunos ejemplos de aplicaciones del aprendizaje supervisado son la aproximación de funciones, clasificación, predicción, etc.
Entre los algoritmos de aprendizaje supervisado se encuentran árboles o reglas de decisión o regresión, clasificación bayesiana, algoritmo de retropropagación en perceptrón multicapa (backpropagation), etc.

- Algoritmos de aprendizaje no supervisado:
  En este caso el sistema también se alimenta de ejemplos, pero no se tiene en cuenta la salida, es decir, no se tiene información acerca de las categorías de esos ejemplos; en vez de esto, se trata de agrupar la información en función de características como la medida de la distancia.
  Algunas aplicaciones: agrupación (clustering), asociación, reducción de dimensionalidad, etc.
  Algunos algoritmos: Estimación de medias (EM), k-medias, mapas auto-organizativos de Kohonen, Máquinas de Soporte Vectorial (SVM), etc.

- Algoritmos de aprendizaje por refuerzo:
  En el aprendizaje por refuerzo, no hay ejemplos de entrenamiento, no se suministran ejemplos etiquetados (aprendizaje no supervisado). El sistema realiza una determinada tarea repetidamente para adquirir experiencia y mejorar su comportamiento, se aprende mediante prueba y error.
  Aplicaciones en procesos que se realizan como una secuencia de acciones: robots móviles (sorteando una serie de obstáculos), ganar un juego de ajedrez, brazo robot (secuencia de movimientos).
  Algunos algoritmos: Q-Learning, Dyna-Q.

Según el tipo de representación de los datos

- Representación proposicional o atributo-valor:
  Cada dato se representa con un número fijo de atributos, junto con la valoración de esos atributos para ese dato.

- Representación en lógica de primer orden o relacional:
  Cada dato se representa con una conjunción de literales que son ciertos para ese dato.

Según el tipo de proceso de aprendizaje

- Aprendizaje Inductivo:
  Es un aprendizaje desde la experiencia: se generaliza a partir de ejemplos de entrenamiento que se reciben como entrada mediante la identificación de características que empíricamente distingan casos positivos de casos negativos. Es la capacidad de obtener nuevos

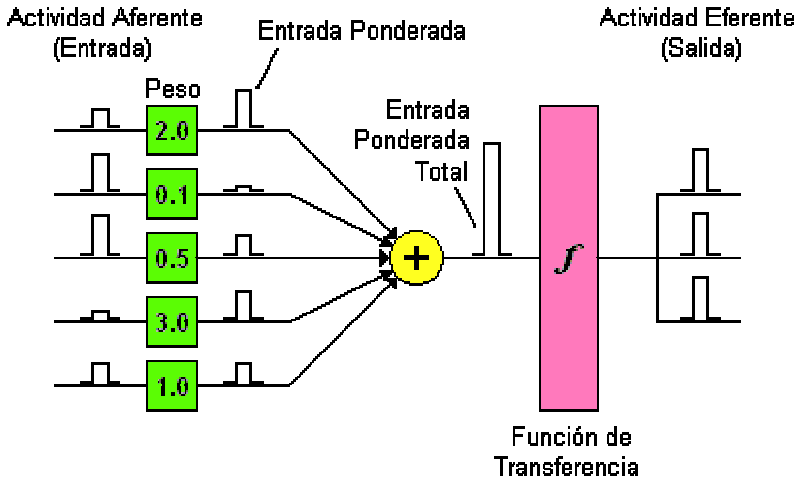
conceptos, más generales, a partir de ejemplos, lo cual conlleva un proceso de generalización / especialización sobre el conjunto de ejemplos de entrada.

- Aprendizaje Deductivo:
  Al igual que el aprendizaje inductivo, el aprendizaje deductivo utiliza ejemplos de entrenamiento, pero además incluye una teoría o conocimiento base que permite explicar o definir una hipótesis que satisface tanto el conjunto de entrenamiento como el conocimiento de base.
  Ejemplos de algoritmos de aprendizaje deductivo: EBL (Explanation Based Learning, Aprendizaje Basado en Explicaciones).

Ejemplos de Algoritmos de Aprendizaje Automático

- *Redes Neuronales Artificiales (ANN)*
  Una red neuronal es un sistema para el tratamiento de la información cuya unidad básica de procesamiento está inspirada en la célula fundamental del sistema nervioso humano, la neurona. Su funcionamiento se basa en las redes neuronales reales, estando formadas por un conjunto de unidades de procesamiento conectadas entre sí. Por analogía con el cerebro humano se denomina *neurona* a cada una de estas unidades de procesamiento. Cada neurona recibe muchas señales de entrada y envía una o varias señales de salida (como ocurre en las neuronas reales). El objetivo es conseguir que el sistema dé respuestas similares a las que es capaz de dar el cerebro, las cuales se caracterizan por su generalización y su robustez.
  Así como el cerebro es capaz de responder a diversas situaciones gracias al aprendizaje recibido mediante ejemplos, las redes neuronales artificiales son capaces de educarse, mediante un entrenamiento realizado a través de reglas de aprendizaje definidas, las cuales están en función tanto de las entradas, como de las salidas (además, mediante la evaluación de las salidas, se pueden ajustar los valores de los pesos que van a afectar las entradas).

Figura 2.17 Proceso de una Red Neuronal Artificial



fuente: HILERA GONZALEZ, José Ramón. Redes Neuronales Artificiales: Fundamentos, Modelos y Aplicaciones.

Las redes neuronales proveen algoritmos de bajo nivel para el aprendizaje y la optimización, que pueden ser combinados con otras tecnologías como lógica difusa o algoritmos genéticos para diseñar sistemas inteligentes.

Dentro de una red neuronal, los elementos de procesamiento se encuentran organizados por capas, en donde una capa es una colección de neuronas. Estas son:

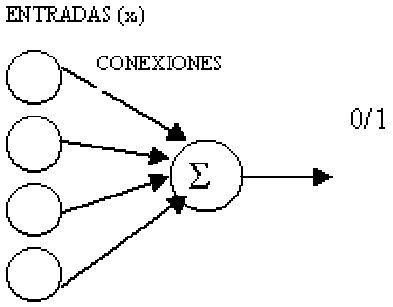
- Capa de Entrada: Recibe las señales de entrada de la red. No siempre se le considera como capa, pues allí no se lleva a cabo ningún proceso.

- Capas Ocultas: No tienen contacto con el medio exterior. Sus elementos pueden tener diferentes conexiones y estas definen la topología de la RNA

- Capa de Salida: Recibe la información de la capa oculta y la transmite al medio exterior.

El modelo más simple de red neuronal es el modelo *perceptron* con una sola neurona. Las neuronas en esta red usan la función de paso como función de activación. El modelo simple de perceptron puede usarse para tomar decisiones binarias. El modelo también puede ser usado como clasificador cuando los ejemplos de entrada son linealmente separables.

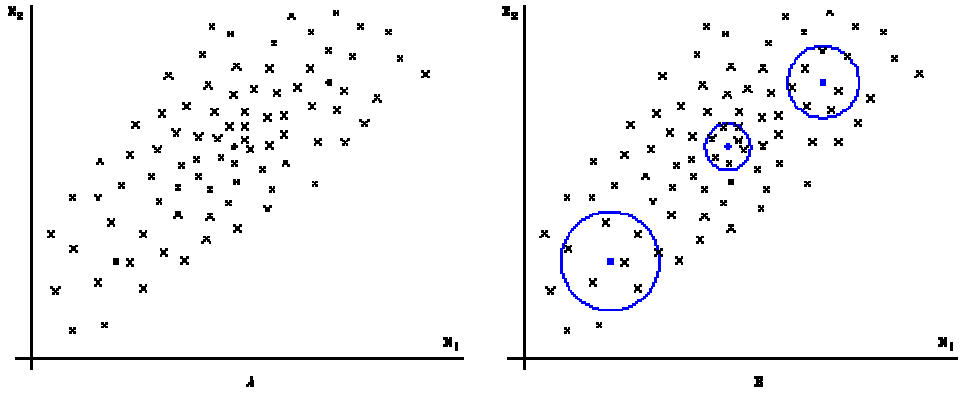
Figura 2.18 Diagrama de una red Perceptron



fuente: HILERA GONZALEZ, José Ramón. Redes Neuronales Artificiales: Fundamentos, Modelos y Aplicaciones.

- *K-Vecinos más Próximos (KNN)*
  Este algoritmo pertenece a la familia de métodos de aprendizaje supervisado. En este algoritmo se almacenan todos los ejemplos del conjunto de entrenamiento y cuando se recibe una nueva instancia x_q, se encuentran los *k* ejemplos de entrenamiento más similares a la nueva instancia. Lo anterior se realiza encontrando la distancia euclidiana entre la instancia y todos los ejemplos de entrenamiento. La distancia euclidiana entre la instancia X_q y un ejemplo x_p se define como $\sqrt{\sum_{i=1}^{m}(a_{x_{qi}} - a_{x_{pi}})}$ donde m es el número de atributos de la instancia y a_{xqi} representa al atributo i de x_q.
  Se eligen los k ejemplos de entrenamiento con las menores distancias. Después de eso se determina la clase de la instancia de acuerdo a la clase más frecuente entre los vecinos (esto se realiza mediante la fórmula $f(x_q) = \arg\max_{v \in V} \sum_{i=1}^{k} w_i d(v, f(x_i))$, donde $w_i = \frac{1}{DE}$ y $d(v, f(x_i)) = \{1,$ si v= $f(x_i)$; 0, si v≠ $f(x_i)$).

Figura 2.19 Método de K Vecinos más Cercanos



- *Regresión Localmente Ponderada (LWR)*

  Este algoritmo, al igual que KNN, pertenece a la familia de algoritmos de aprendizaje basados en instancias. En este algoritmo se almacenan todos los ejemplos del conjunto de entrenamiento y cuando se recibe una nueva instancia x_q, se encuentran los ejemplos de entrenamiento más similares a la nueva instancia los cuales servirán para clasificarla. LWR construye una aproximación explícita de la función objetivo *f* sobre una región que rodea al punto x_q . Dado un punto x_q, para predecir sus parámetros de salida y_q, se asignan a cada punto de entrenamiento un peso dado por la distancia inversa del punto x_q al punto de entrenamiento: $w_i = \frac{1}{|x_q - x_i|}$.

  Sea *W* , la matriz de pesos, una matriz diagonal con entradas $w_1,...,w_n$. Sea *X* una matriz cuyas filas son los vectores $x_1,...,x_n$, los parámetros de entrada de los ejemplos en el conjunto de entrenamiento con un "1" adicional en la última columna. Sea *Y* una matriz cuyas filas son los vectores $y_1,...,y_n$, los parámetros de salida de los ejemplos en el conjunto de entrenamiento. Entonces los datos de entrenamiento ponderados están dados por $Z = WX$ y la función objetivo ponderada es $V = WY$. Entonces se usa el estimador para la función objetivo definido como $y_q = x_q^T Z^* V$ donde Z* es la pseudoinversa de Z.

  LWR es normalmente aplicado a problemas de regresión, pero es fácilmente adaptable a problemas de clasificación. Para un problema de clasificación con n clases, se provee como parámetro de salida para cada ejemplo un vector de n elementos, donde el i-ésimo elemento del vector es 1 si el ejemplo pertenece a la clase i y cero en caso contrario. La clase de un ejemplo de prueba se determina por el valor más alto del vector de salida.

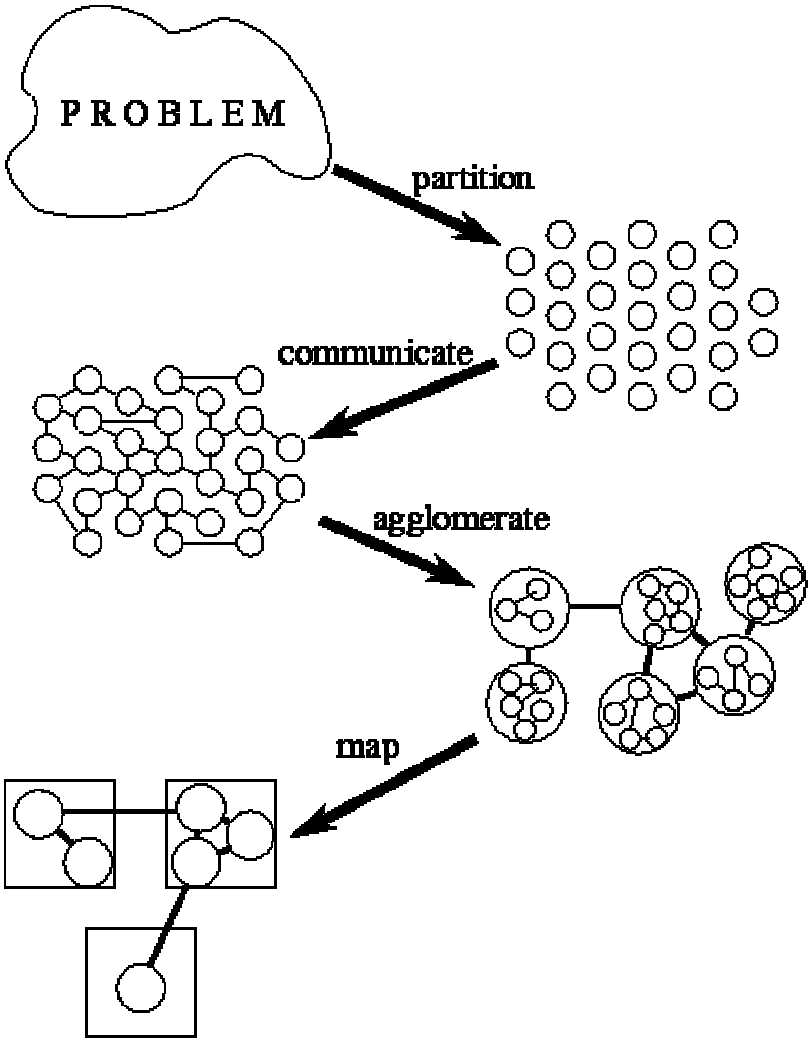
2.5 DISEÑO DE ALGORITMOS PARALELOS

La mayoría de problemas de programación tienen soluciones paralelas. La mejor solución secuencial puede ser a menudo la peor solución paralela.
La metodología en que se basa el diseño de algoritmos paralelos comprende 4 etapas: partición, comunicación, aglomeración y mapeo, las cuales son mostradas a continuación:

Tabla 2.2 Etapas de la metodología del diseño paralelo

ETAPA	DESCRIPCION
Particionamiento	La tarea computacional y los datos son divididos en pequeñas tareas. En esta etapa el numero de procesadores es ignorado, la atención se centra en las oportunidades de paralelismo
Comunicación	La comunicación requerida para coordinar la ejecución de tareas es determinada, y se escoge las estructuras apropiadas de comunicación, y los algoritmos son definidos.
Agrupación	Las tareas y estructuras de comunicación definidas en las dos etapas anteriores son evaluadas con respecto a los requerimientos de rendimiento y los costos de implementación. Si es necesario, las tareas son combinadas para aumentar el rendimiento y reducir la comunicación.
Asignación	Cada tarea es asignada a cada procesador de manera que se satisfagan las metas de máxima utilización de procesamiento y mínimos costos de comunicación

Figura 2.20 Etapas de la metodología del diseño paralelo



fuente: FOSTER, Ian. Designing and Building Parallel Programs

Particionamiento:
La fase de partición es entendida como buscar las oportunidades de ejecución paralela. Entonces la idea es definir una gran cantidad de tareas pequeñas para hacer una *granularidad fina*. Cuando la granularidad es fina es más fácil apilar las pequeñas tareas. Una descomposición de grano fino provee una excelente flexibilidad en términos de aumentar el potencial de paralelización. En las siguientes fases, de acuerdo a la evaluación de requerimientos de comunicación, la arquitectura o software de desarrollo, se descartaran algunas oportunidades de paralelización.

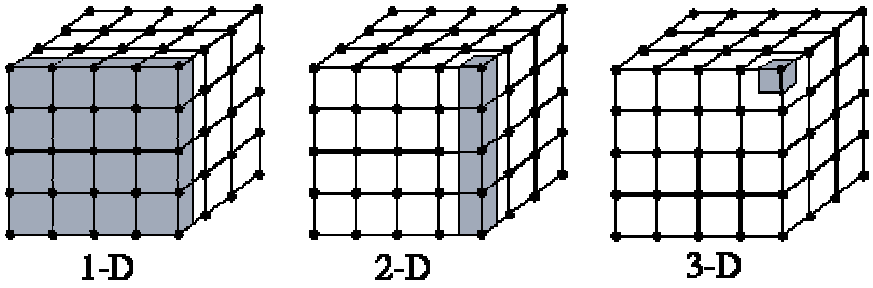
Una buena paralelización divide en pequeñas piezas: se realiza división de instrucciones de máquina o división de los datos. Cuando se realiza una descomposición del los datos asociados con el problema se denomina *descomposición de dominio*. Otra técnica alternativa, primero descomponer el número de tareas, y después descomponer los datos es llamada *descomposición funcional*.

Descomposición de Dominio:
Con la descomposición de datos se da una aproximación a un problema de particionamiento, lo primero que se busca es la descomposición de los datos asociados al problema. Si es posible se busca dividir en pequeñas piezas del mismo tamaño. Después que la partición esta hecha, se asocia cada operación con los datos con los cuales va a operar. Esta partición produce un número de tareas, cada una restringida a los datos sobre los cuales va a operar. Una tarea puede requerir datos de diferentes tareas. En ese caso se necesita un canal para comunicar las diferentes tareas. Este requerimiento se analiza en la siguiente etapa.

Los datos pueden ser distribuidos en la entrada del problema, en la salida o mientras se realiza el proceso. Se puede realizar diferentes tipos de particiones, basado en diferentes estructuras de datos. Se sugiere que se empiece por la estructura de datos principal, o a la que se tenga más acceso a lo largo del programa.

La figura 2.21 muestra la descomposición de un problema que tiene una grilla de 3 dimensiones. Las instrucciones son ejecutadas en cada punto de la grilla repetidamente. La descomposición en el eje x, y o z se puede realizar de la forma mas fácil, la descomposición más agresiva en este caso sería, en cada punto de la grilla. Cada tarea mantendría en un estado varios valores asociados al punto de la grilla y es responsable por el cómputo que sea necesario para actualizar su estado.

Figura 2.21 Descomposición de dominio para un problema que involucra una grilla de tres dimensiones. Es posible una descomposición de una, dos o tres dimensiones; en cada caso los datos asociados con una tarea es sombreado.



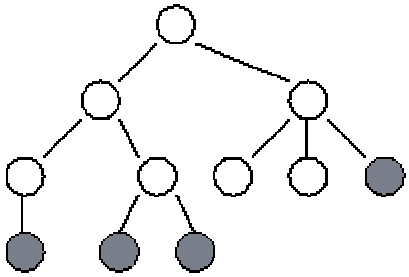
fuente: FOSTER, Ian. Designing and Building Parallel Programs

Descomposición Funcional

La descomposición funcional ofrece una alternativa diferente y complementaria. En esta aproximación la atención se centra en la computación que será ejecutada sin pensar en los datos que se utilizarán. Después de realizar la división en diferentes tareas, se procede a revisar los datos que se necesitarán para realizarla. Los requerimientos de datos pueden ser diferentes para cada tarea, en ese caso la partición ha sido completa. De lo contrario aparecen solapamientos de datos, en este caso será necesaria una cantidad de comunicación considerable. Esta es una razón por la cual se prefiere realizar una descomposición de datos.

Uno de los ejemplos donde la descomposición funcional puede ser la más apropiada es una exploración en un árbol para la búsqueda del nodo "soluciones". La descomposición del dominio seria obvia. Sin embargo una descomposición de grano fino puede ser obtenida: Inicialmente una tarea simple es creada en el nodo raíz del árbol, la tarea evalúa ese nodo y si no es que se necesita crearía una nueva tarea (subárbol). Una nueva tarea será creada cada vez que se expanda la búsqueda en el árbol (ver Figura 2.22)

Figura 2.22 Estructura de tarea para un ejemplo de búsqueda. Cada circulo representa un módulo en el árbol de búsqueda y una llamada a la tarea nueva. Las líneas representan los canales donde retorna la solución.



Comunicación:
Las tareas generadas en la partición son pensadas para ejecutarse concurrentemente, pero en general no son independientes. Las tareas generalmente requieren datos de otras. Los datos deben ser transmitidos de una tarea a otra para que pueda proseguir con su cómputo. Este flujo de información debe ser especificado en la etapa de comunicación.

Entonces surge la necesidad de conceptuar comunicación entre tareas, como un canal que las une, en el cual una tarea envía mensajes y otra la recibe. Entonces la comunicación asociada con un algoritmo puede ser especificada en 2 partes:
- La primero define un canal que une, directa o indirectamente, tareas que requieren datos (consumidor) , con las tareas que poseen esos datos (productor)
- En la segunda se especifica los mensajes que serán enviados y recibidos por ese canal.

La comunicación se puede clasificar como:
- Comunicación local: cada tarea se comunica con un pequeño grupo de tareas “vecinas”, en contraste la comunicación global que requiere que cada tarea se comunique con un gran número de tareas.
- Comunicación estructurada: la tarea y sus vecinos forman una estructura regular: como un árbol o una grilla, mientras que en la comunicación irregular la comunicación pueden ser grafos arbitrarios.
- En la comunicación sincrónica productores y consumidores ejecutan la tarea en forma coordinada, mientras que en la comunicación asincrónica el consumidor obtiene los datos sin ayuda del productor.

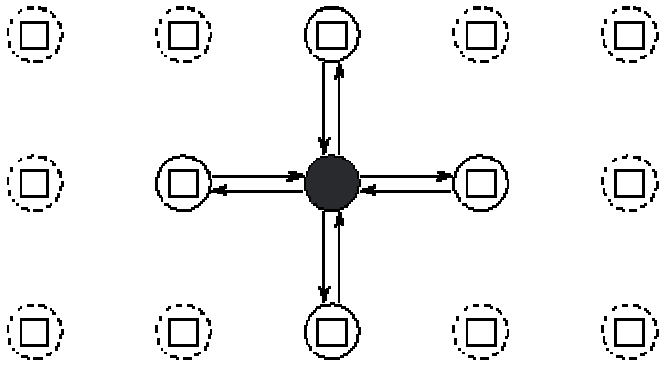
Comunicación local
Una estructura de comunicación local es obtenida cuando una operación necesita datos de un pequeño número de tareas Cuando este tipo de comunicación ocurre se debe crear un canal entre el consumidor y productor de tareas, e introducir operaciones de recibo y envío entre las tareas.

Por ejemplo, los requerimientos de comunicación asociados con un método numérico simple: El método de diferencias finitas de Jacobbi. La siguiente expresión se usa para actualizar cada uno de los puntos X_{ij} de la grilla X

$$X_{i.j}^{t+1} = \frac{4X_{i.j}^{t} + X_{i.-1.j}^{t} + X_{i+1.j}^{t} + X_{i\ j-1}^{t}\ X_{i\ j+1}^{t}}{8}$$

Esta actualización es aplicada repetidamente cuando $t=1,2,3,\ ..\ ,\ n$ .La notación $X_{i.j}^{t}$ simboliza el valor del punto de la grilla $X_{i.j}$ en el paso de tiempo t.

Figura 2.23. Tarea y estructura del canal en una grilla de dos dimensiones que aplica el método de diferencias finitas con 5 puntos. Solo son mostrados los canales en los cuales la tarea esta involucrada



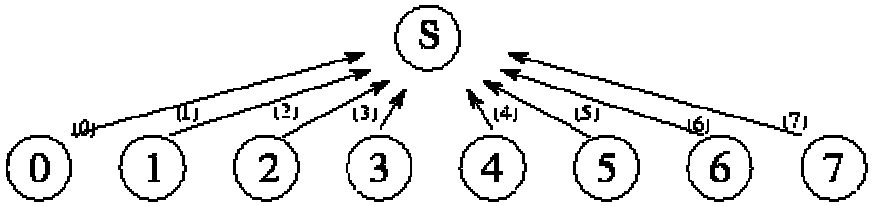
fuente: FOSTER, Ian. Designing and Building Parallel Programs

Comunicación global
Una operación de comunicación global es aquella en la que varias tareas tienen que participar. Cuando cada operación es implementada, es difícil distinguir cada par de consumidores y productores.
Por ejemplo considere que se necesita recoger los resultados de N operaciones realizadas en N tareas usando el operador de la suma.

$$S=\sum_{i=0}^{n-1} = X_i$$

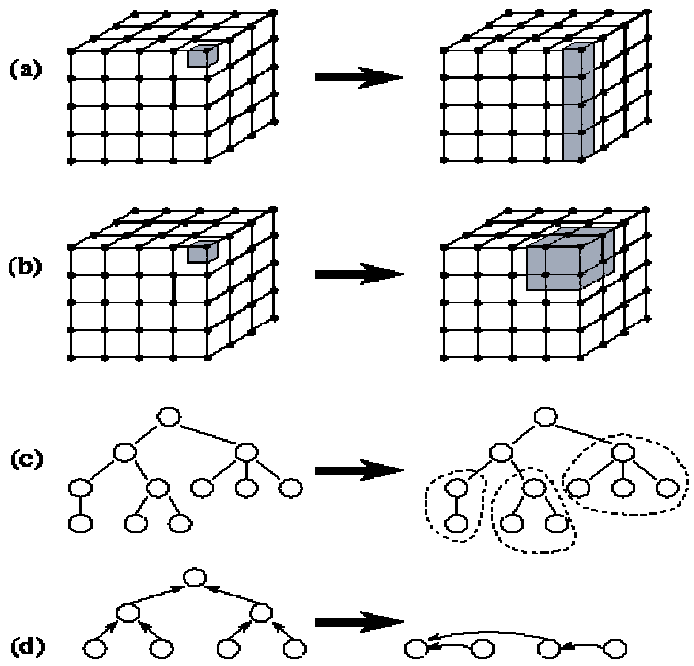
Se asume una tarea líder que requiere el resultado de S. Desde el punto de vista local, se deben crear canales independientes en donde cada tarea 0,1., … , n envía su X_i a la tarea líder que debe acumular cada uno de los recibos en la variable S, como se muestra en la figura 2.24.

Figura 2.24. Un algoritmo centralizado de suma que usa una tarea líder S para sumar N números distribuidos a lo largo de n tareas. En este ejemplo N=7 cada uno de los canales esta etiquetado con el número del canal



Agrupación:
En las primeras dos fases del proceso de diseño, se realiza la partición de la computación a ser ejecutada en pequeñas tareas y se introduce la computación requerida por esas tareas. El algoritmo resultante, es abstracto debido a que no esta especializado en una eficiente ejecución para una máquina paralela en especial. Este factor puede hacerlo ineficiente.
En esta tercera etapa, la agrupación, se mueve de lo abstracto a lo concreto. Se revisa si el algoritmo que se obtuvo en las 2 fases anteriores se adecua a la maquina paralela donde se piensa implementar.
El número de tareas agrupadas en esta fase, que deben tender a la reducción, deben ser más grande que el numero de procesadores a utilizar. Alternativamente, durante la etapa de agrupación se puede reducir el número de tareas al número de procesadores a utilizar (Figura 2..25)

Figura 2.25 Ejemplos de agrupación: a) El tamaño del trabajo es incrementado para reducir la dimensión de descomposición de tres a dos. B) Las tareas adyacentes son combinadas para lograr una descomposición de tres dimensiones de mayor granularidad. C) subárboles en una estructura divide y vencerás son agrupados. d) Nodos de un árbol son combinados



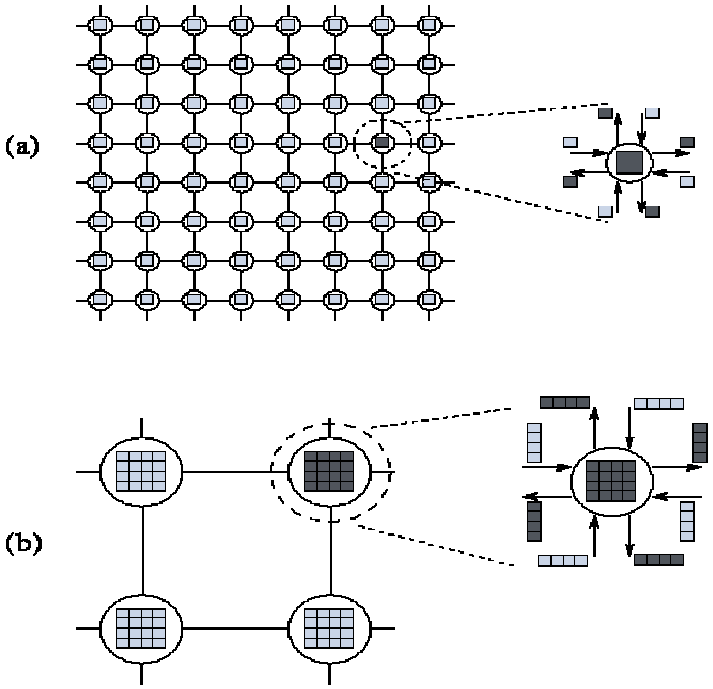
fuente: FOSTER, Ian. Designing and Building Parallel Programs

Tres factores influyen a la hora de agrupar: granularidad, flexibilidad y Costos del Software.

Incrementar Granularidad:
Una parte crítica que influencia el rendimiento de un algoritmo paralelo son los costos de comunicación. En la mayoría de maquinas paralelas, se tiene que parar la computación, para recibir y enviar mensajes.
Para resolver este problema se pude tomar dos caminos:

- Mejorar el rendimiento enviando menos datos.
- Disminuir el número de envíos realizados pero con la misma cantidad de dato

Figura 2.26 Costos de comunicaciones



fuente: FOSTER, Ian. Designing and Building Parallel Programs

La figura 2.26 muestra el costo de comunicación al hacer más grande la granularidad en un problema de diferencias finitas en 2D.
En la parte a) se muestra una grilla de 8*8 que es particionada en 8*8=64 tareas, cada una responsable por un punto, y hay un total de 64*4=256 comunicaciones, 4 por cada tarea.
Mientras que en la parte b) la grilla es dividida en 2*2=4 tareas cada una responsable por 16 puntos, y un total de 4*4=16 comunicaciones, 16*4=64 valores transferidos por tarea.

Preservar la Flexibilidad:
La habilidad de crear tareas variables es crítica si se quiere que el programa sea escalable y portable. Esta flexibilidad es útil para adecuar el código para un computador en particular.
Un beneficio de crear más tareas que procesadores es que permite tener una gran variedad en la siguiente etapa de asignación, para distribuir la carga.

Reduciendo costos de Ingeniería del Software
Hasta esta parte se ha asumido, que se debe escoger la mejor agrupación para mejorar la eficiencia y flexibilidad del algoritmo. Una consideración adicional,

cuando existen códigos secuenciales, es el costo relativo asociado con las diferentes técnicas de partición. Desde este punto de vista se puede escoger no realizar cambios significativos. Por ejemplo, si el código opera un grid multidimensional, puede ser ventajoso preservar las particiones alrededor de una dimensión, si este posee rutinas, que no puedan ser cambiadas en el programa paralelo.

Asignación

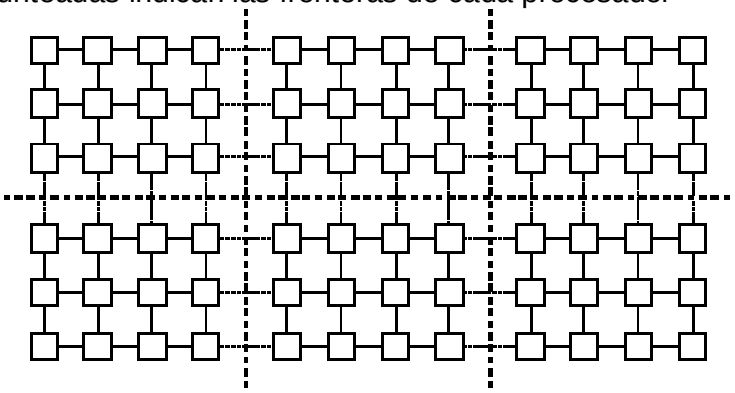
La cuarta y ultima etapa del proceso de diseño es la asignación, donde se especifica donde se va a ejecutar cada tarea. En esta clase de computadores, se requiere, que el grupo de tareas y requerimientos de comunicación se encuentren bien especificados en el algoritmo paralelo; sistemas operativos o mecanismos de hardware, puede ayudar a asignar las tareas ejecutables en los procesadores disponibles. Desafortunadamente, los mecanismos de mapeo deben ser realizados para poder ser ejecutados en computadores paralelos escalables. En general, la asignación es una tarea difícil que debe ser realizada explícitamente en el diseño de algoritmos paralelos.

La meta al desarrollar algoritmos paralelos es minimizar el tiempo total de ejecución. Se pueden usar dos estrategias para lograrlo:

- Asignar tareas que se pueden ejecutar concurrentemente en diferentes procesadores para evitar la concurrencia
- Asignar tareas que se comuniquen frecuentemente en el mismo procesador para incrementar la comunicación entre procesadores.

Además las limitaciones de recursos pueden restringir el número de tares asignadas a un procesador.

Figura 2.27. Asignación de un problema de grilla en el cual cada tarea ejecuta la misma cantidad de computación y comunicación solo con cuatro vecinos. Las líneas punteadas indican las fronteras de cada procesador



Muchos algoritmos desarrollados que usan descomposición de dominio usan la técnica de fijar tareas de igual tamaño, comunicación local estructurada y global. En estos casos una asignación eficiente es sencilla. Este mapeo de tareas minimiza la comunicación entre procesadores (Figura 2.27); también permite agrupar tareas en el mismo procesador, esto produce un total de P tareas de grano grueso, una por procesador.

Un algoritmo más complejo que se base en la descomposición de dominio en el cual se trabaje por tarea y / o la comunicación no sea estructurada, la agrupación y descomposición de dominio no es tan obvia para el programador. Entonces allí se utiliza algoritmos de *balanceo de carga* que buscan identificar estrategias eficientes de agrupación y asignación, usualmente usan técnicas heurísticas. El tiempo requerido para ejecutar estos algoritmos debe ser "pesado" con el fin de ganar en tiempo de ejecución.

Los problemas más complejos son aquellos en los cuales el número de tareas o la cantidad de comunicación o computación cambia dinámicamente durante el tiempo de ejecución. En ese caso, para solucionar el problema de descomposición de dominio se puede usar la estrategia de *balanceo dinámico de carga*, con la cual un algoritmo es ejecutado periódicamente para determinar la nueva aglomeración y asignación.

Los algoritmos basados en descomposición funcional producen frecuentemente tareas que coordinan a otras tareas al inicio y final de la ejecución. En ese caso se pueden usar algoritmos de *programación de tareas*, que asignan tareas a un procesador que se encuentren ociosos.

3. IDENTIFICACIÓN DE GALAXIAS: DESARROLLO SECUENCIAL

3.1 METODOLOGÍA DE TRABAJO

Analizados en detalle distintas metodologías de desarrollo y teniendo en cuenta las características del proyecto, se eligió el modelo de *Prototipado evolutivo* como una aproximación a la solución.

Las características y las necesidades del proyecto son:

- Los requerimientos y especificaciones del proyecto solo están descritos globalmente al inicio de éste.
- Las especificaciones deben poder ajustarse a posibles modificaciones y adaptaciones que se hagan en el transcurso del desarrollo del proyecto.

Dado esto, se muestran las ventajas del modelo de Prototipado Evolutivo, las cuales se ajustan y dan solución a las necesidades que presenta el proyecto:

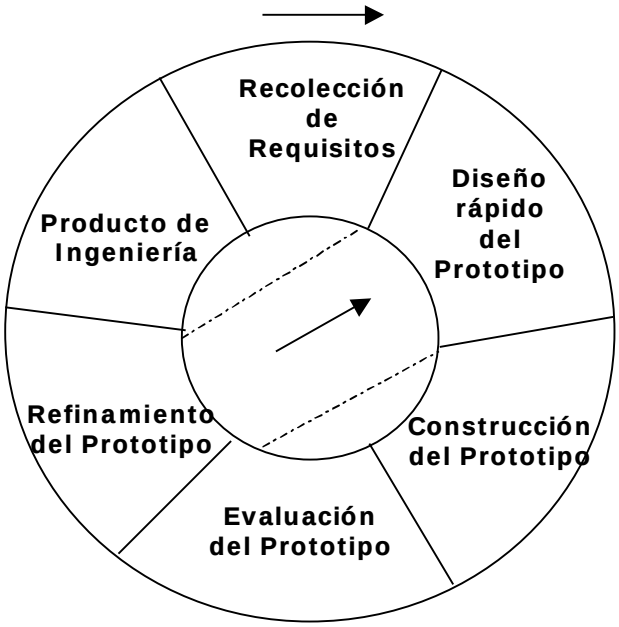
- El modelo no exige una fuerte planificación.
- Se pueden realizar cambios en etapas tempranas y así emitir varios prototipos evaluables durante el desarrollo, obteniéndose paralelamente una metodología integral para el proceso de evaluación del sistema.
- Funciona con incrementos, los cuales arrojan como resultado un prototipo, para luego agregar nuevas funcionalidades hasta cumplir los objetivos propuestos.
- Propicia un intercambio de conocimientos y de autocrítica al sistema, lo que conlleva a que se produzcan muchas pruebas antes de liberar una nueva versión así como mejoras rápidas a problemas que puedan surgir durante su uso.
- Genera signos visibles de progreso.

Así, la construcción de prototipos comienza con la recolección de los requisitos, después se definen los objetivos globales del sistema, se identifican todos los requisitos conocidos y se perfilan las áreas en donde será necesaria una mayor definición.

Luego se produce un "diseño rápido" que conduce a la construcción de un prototipo el cual es evaluado y se utiliza para refinar los requisitos del software a desarrollar.

Después se realiza un proceso interactivo en el que el prototipo es "refinado" para que satisfaga las necesidades del usuario, al mismo tiempo que facilita al desarrollador una mejor comprensión de lo que hay que hacer y poder entregar el producto final requerido.

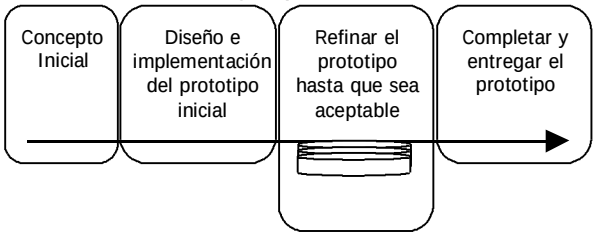
Figura 3.1. Modelo de Prototipado Evolutivo



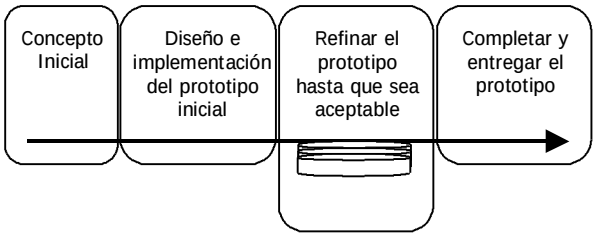
fuente: PRESSMAN, Roger. Ingeniería del Software: Un Enfoque Práctico.

En el proyecto se pueden distinguir dos etapas de prototipado, así:

Figura 3.2 Etapas de la metodología planteada



ETAPA DONDE, MEDIANTE LA METODOLOGÍA DE PROTOTIPADO SE BRINDA UNA SOLUCION SECUENCIAL



ETAPA DONDE, MEDIANTE LA METODOLOGÍA DE PROTOTIPADO SE DISEÑA UNA SOLUCION PARALELA

En la primera etapa, usando la metodología de prototipado evolutivo y de acuerdo a los objetivos planteados analizamos, diseñamos, implementamos y entregamos una solución de tipo secuencial, la cual nos permita tomar la imagen de una galaxia, mediante algoritmos extraer los parámetros que la definen, ingresar estos a un algoritmo de aprendizaje automático y finalmente identificar la clase de la galaxia.

En la segunda etapa nuevamente mediante la metodología de prototipado evolutivo tomamos la solución secuencial, la analizamos, definimos requisitos y diseñamos la solución paralela buscando que el proceso se desarrolle de una forma más rápida y eficiente.

3.2 PLAN DE TRABAJO

De acuerdo a la metodología planteada anteriormente, se definieron las siguientes fases como componentes del plan de trabajo:

- *Fase Inicial:*
  En esta fase se tendrán en cuenta las siguientes etapas:

  Recopilación de Información: Esta etapa comprende la actividad de recolección del material necesario, como soporte durante el desarrollo del proyecto. Este material son los artículos, libros, páginas web, etc.

  Entrenamiento Previo: En esta etapa se lleva a cabo la familiarización con los sistemas operativos, lenguajes de programación (Matlab), herramientas de paralelización y los conceptos teóricos de morfología de galaxias, tratamiento de imágenes, algoritmos de aprendizaje automático y procesamiento paralelo.

  Análisis y Especificación: En esta etapa se realiza una recopilación y análisis de los requerimientos del sistema:
  Se requiere una base de datos de imágenes de galaxias, en distintos formatos, de diferentes fuentes (surveys). Se requiere preferiblemente un número grande de imágenes para tener más confiabilidad en los resultados de las pruebas.
  Se requieren imágenes estandarizadas en cuanto al tamaño (píxeles), el formato y la orientación de los ejes principales de las galaxias para tener una homogeneidad en la extracción de parámetros.
  Se requiere expresar las imágenes en forma de ejemplos de entrenamiento (vectores de parámetros) del algoritmo de aprendizaje automático.
  Se requiere un entendimiento claro y detallado del proceso secuencial, para definir los parámetros del diseño del algoritmo paralelo.

- *Fase iterativa:*

  En esta fase se empieza con la realización del prototipo inicial, teniendo en cuenta las especificaciones de los requerimientos de la fase anterior; posteriormente se realiza la actividad de refinar este prototipo por medio de las siguientes actividades:

  Diseño: En esta etapa se realiza la selección de los algoritmos para el preprocesado de la imagen, la identificación de los parámetros que caracterizan una galaxia, la selección de los algoritmos de extracción de dichos parámetros, el diseño del algoritmo de aprendizaje automático y la forma de paralelizar todo el proceso.

  Desarrollo: En esta etapa se realiza la implementación del prototipo definido para el ciclo actual, basándose en el diseño definido en la etapa anterior: Se aplica el Tratamiento digital de imágenes para realizar el preprocesado y las técnicas de extracción de parámetros y se aplican los algoritmos de aprendizaje automático.

  Plan de pruebas: Se realizan las pruebas necesarias para obtener indicadores cualitativos y cuantitativos del funcionamiento del sistema.

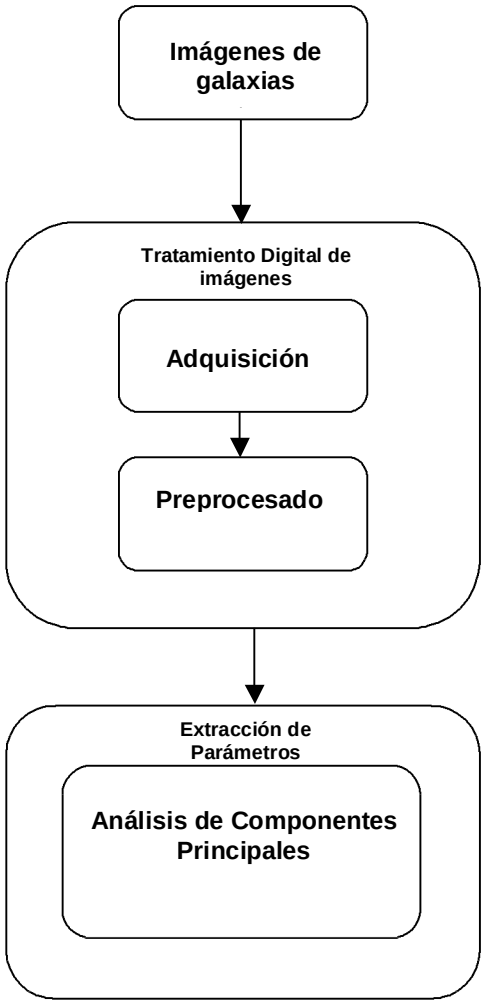
  Análisis de resultados: En esta etapa se realiza una evaluación de los resultados obtenidos de la etapa anterior, para decidir si se avanza a la siguiente fase, o se vuelve a la fase iterativa.

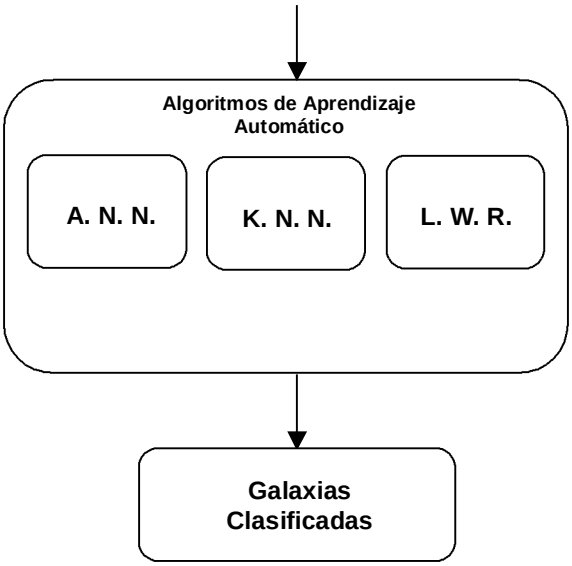
  Por último se plantea el diseño del algoritmo en paralelo.

3.3 DESARROLLO SECUENCIAL

El siguiente diagrama muestra las fases del desarrollo secuencial.

Figura 3.3 Etapas del desarrollo secuencial





3.3.1 Tratamiento Digital de la Imagen

3.3.1.1 Adquisición:

El conjunto de imágenes usadas en este trabajo fueron tomadas de diferentes fuentes:

- SDSS (Sloan Digital Sky Survey), http://www.sdss.org/
  Las imágenes están en formato ".jpg", ".gif", ".tiff"

- NED (NASA/Ipac Extragalactic Database), http://nedwww.ipac.caltech.edu/
  Imágenes en diferentes longitudes de onda: infrarrojo (desde 1000 μ hasta 2.5 μ), JHK (de 2.2 μ hasta 1.2 μ) y azules (~ 0.440 μ). Estas imágenes están en formato ".fit" que es un formato estándar para las imágenes astronómicas.

- Hubble Space Telescope (Telescopio Espacial Hubble), http://hubblesite.org
  Imágenes en diferentes formatos: ".jpg", ".gif" ".tiff"

- NOAO (National Optical American Observatory, Observatorio Nacional Óptico Americano) http://www.noao.edu

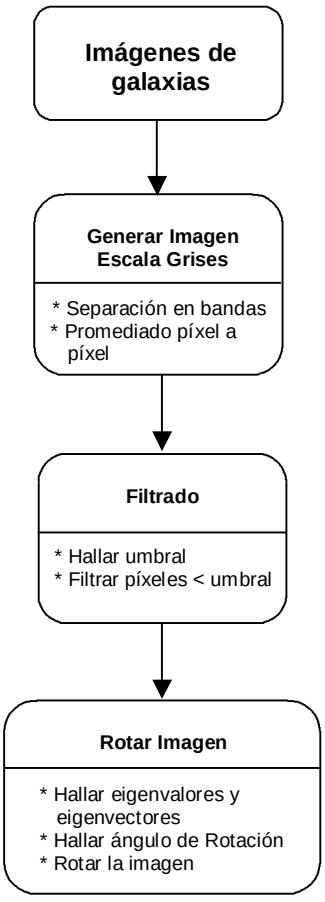
Imágenes en diferentes formatos: “.jpg”, “.png”, “.tiff”

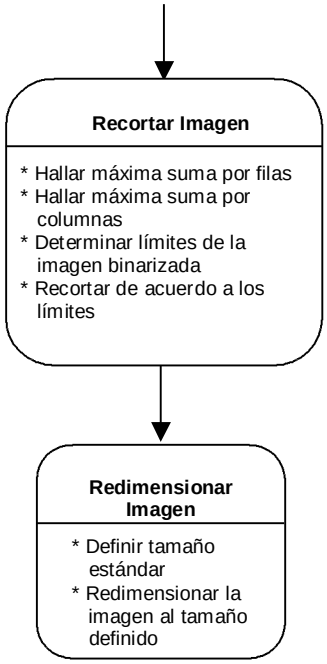
- NASA Photo Journal, http://photojournal.jpl.nasa.gov
  Imágenes en diferentes formatos: “.jpg”, “.png”, “.tiff”

Se trabajó únicamente con imágenes “.jpg”, ya que se necesitó una estandarización en las imágenes a trabajar. Las imágenes en otros formatos se convirtieron a “.jpg”.

3.3.1.2 Preprocesado:
Estas son las etapas principales en el preprocesado.

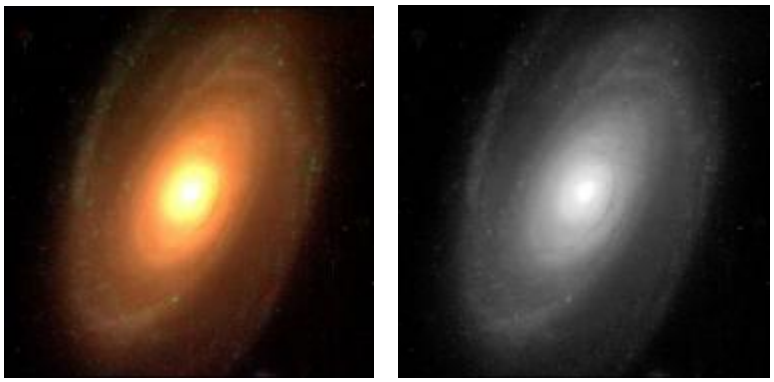
Figura 3.4 Etapas del preprocesado





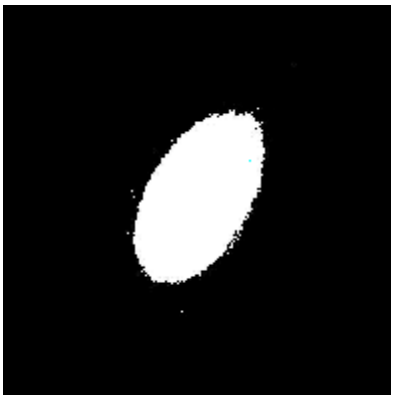
Después de adquiridas las imágenes, el primer paso del preprocesamiento es aplicar una operación de mejoramiento por procesamiento de punto, es decir, tratando los píxeles individualmente: se convierte la imagen en color a una imagen en escala de grises, para lo cual se realiza un promedio de intensidades entre las componentes R, G y B.

Figura 3.5 Imagen de la Galaxia Espiral NGC3031 Original (izquierda) y Promediada (derecha)



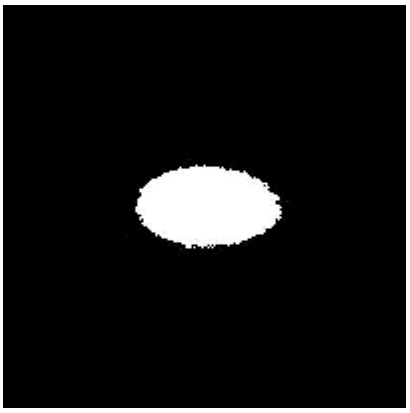
A continuación se binariza la imagen (también es una operación de mejoramiento por procesamiento de punto), lo cual se hace filtrando todos los píxeles que tengan una intensidad inferior a cierto valor (llamado umbral, que se define de acuerdo al rango de intensidades de la imagen), así se obtiene una imagen donde, todos los píxeles tienen una intensidad mayor al valor mencionado.

Figura 3.6 Imagen de la Galaxia Espiral NGC3031 Binarizada



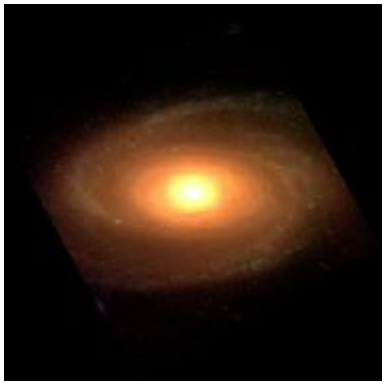
Se pretende una homogeneidad en la simetría de las imágenes, es decir, que todas queden dispuestas horizontalmente (que el eje mayor de la imagen de la galaxia esté ubicado de manera horizontal). Para tal caso, se encuentra el ángulo de rotación y se le aplica a la imagen binarizada para que ésta quede horizontalmente, tal y como se espera.

Figura 3.7 Imagen de la Galaxia Espiral NGC3031 Binarizada y Rotada



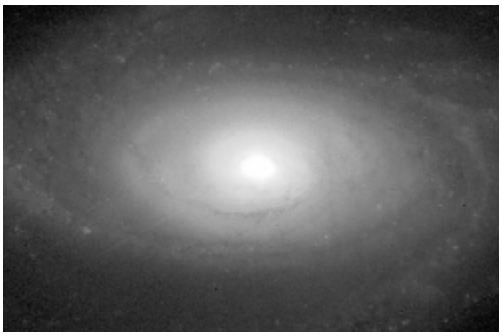
Hasta ahora se ha aplicado las operaciones a la imagen a escala de grises y luego a la binarizada, sin embargo, se puede aplicar la rotación a la galaxia original, pues ya se conoce el ángulo de rotación.

Figura 3.8 Imagen de la Galaxia Espiral NGC3031 Original Rotada



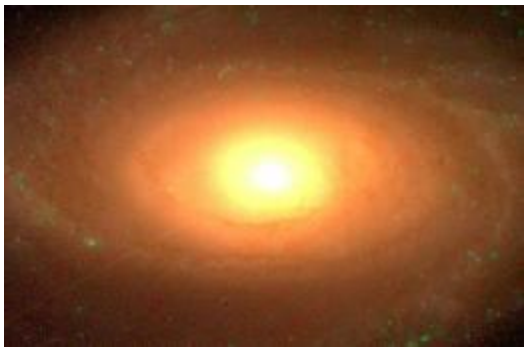
El último paso del preprocesado es recortar la imagen, para que quede únicamente la parte de la galaxia, y ajustar la imagen a un tamaño estándar (buscando homogeneidad en el tamaño de las imágenes de las galaxias).

Figura 3.9 Imagen de la Galaxia Espiral NGC3031 Promediada, Rotada y Recortada



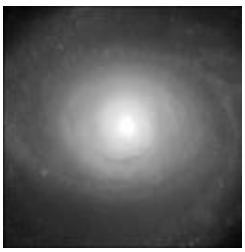
Igualmente se aplica para la imagen original

Figura 3.10 Imagen de la Galaxia Espiral NGC3031 Original, Rotada y Recortada



El tamaño original de las imágenes es diferente en cada una de ellas, pero ahora se aplica la última operación del preprocesado, ajustar la imagen a un tamaño estándar (128 x 128 píxeles en nuestro caso).

Figura 3.11 Imagen de la Galaxia Espiral NGC3031 Promediada, Rotada, Recortada y Ajustada

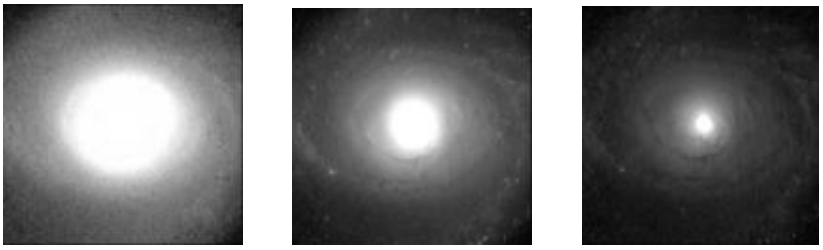


Igualmente para la imagen original

Figura 3.12 Imagen de la Galaxia Espiral NGC3031 Original, Rotada, Recortada y Ajustada

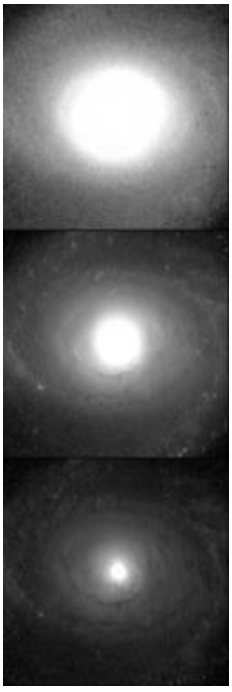
Ahora, y antes de pasar a la etapa de extracción de parámetros, separamos la imagen en sus bandas R, G, y B.

Figura 3.13 Imagen de la Galaxia Espiral NGC3031 en sus componentes R (izquierda), G (centro) y B (derecha).



Tenemos dos alternativas: trabajar con las imágenes por separado o unirlas en una sola imagen (de tamaño 384 x 128 píxeles)

Figura 3.14 Imagen de la Galaxia Espiral NGC3031 en sus componentes R, G y B combinadas.

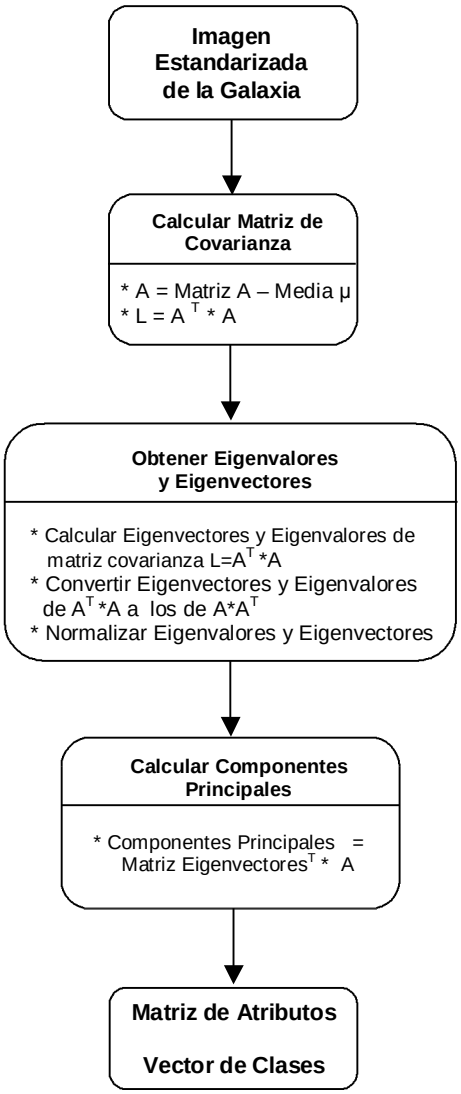


En este punto la imagen está lista para la siguiente etapa, la extracción de parámetros.

3.3.2 Extracción de Parámetros

Estas son las etapas principales en la aplicación de los algoritmos de extracción de parámetros

Figura 3.15 Etapas de la aplicación de algoritmos de extracción de parámetros



Esta etapa empieza con la entrada de las imágenes estandarizadas.
Dentro del trabajo pretendemos probar las diferentes formas en que se puede presentar la imagen de una galaxia y evaluar cual es la mejor de acuerdo a los resultados obtenidos; entonces se muestra que una imagen puede ser trabajada de tres formas:

Imagen en escala de grises (una banda)

Figura 3.16 Imagen en escala de grises

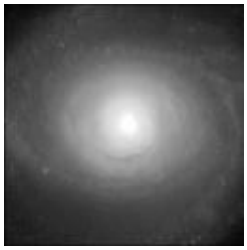


Imagen a color (tres bandas R,G,B)

Figura 3.17 Imagen en color

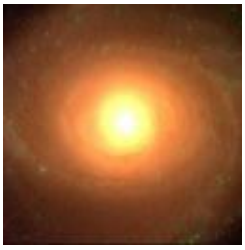
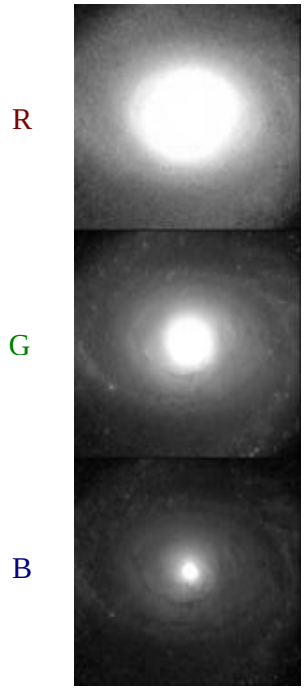
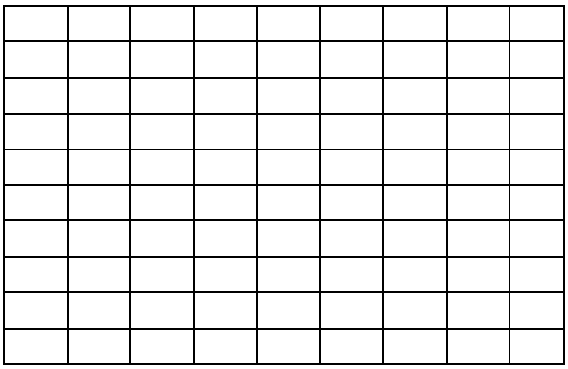


Imagen Compuesta por las bandas separadas R, G y B

Figura 3.18 Combinación de bandas



Ahora, independientemente de la forma en que se presente la imagen, esta sigue siendo una imagen, pero para nuestro trabajo se debe trabajar como una matriz donde cada casilla equivale a un valor de intensidad. Esta imagen de entrada será llamada A (m*n).



Una vez tengamos nuestra matriz (imagen) debemos hallar la matriz de covarianza que es muy importante pues nos indica la variación existente entre los diferentes puntos de la imagen.

Para esto calculamos la media para los valores de cada fila y después restamos esos valores a la matriz original.

$$\mathbf{A = A(i) - \mu(i)}$$

Después, haciendo un análisis sencillo podemos concluir que es mejor trabajar con una matriz de n*n que con una de m*m (sabiendo que m>n), por eso calcularemos la matriz de varianza – covarianza como A^T*A y no como $A*A^T$ $\mathbf{L = A^T*A}$

Encontramos los eigenvalores y eigenvectores para la matriz de covarianza L. Obtenemos dos matrices: una donde cada columna es un eigenvector V_i de la matriz de covarianza y otra matriz donde los valores de la diagonal λ_i son los eigenvalores de L. Cada eigenvalor esta directamente relacionado con un eigenvector.

$$V = \begin{pmatrix} v_{11} & v_{12} & v_{13} \ldots\ldots\ldots\ldots v_{1n} \\ v_{21} & v_{22} & v_{23} \ldots\ldots\ldots\ldots v_{2n} \\ & & \\ & & \\ v_{n1} & v_{n2} & v_{n3} \ldots\ldots\ldots\ldots v_{nn} \end{pmatrix} \quad \lambda = \begin{pmatrix} \lambda_1 & 0 & 0 & \ldots\ldots\ldots\ldots 0 \\ 0 & \lambda_2 & 0 & \ldots\ldots\ldots\ldots 0 \\ & & & \\ & & & \\ 0 & 0 & 0 & \ldots\ldots\ldots\ldots \lambda_n \end{pmatrix}$$

Ahora, como encontraron los eigenvectores para A^T*A se deben transformar para que correspondan a los eigenvectores de $A*A^T$

$$\mathbf{V = A * V}$$

Normalizo los eigenvectores dividiendo cada vector columna por su norma y normalizo los eigenvalores dividiéndolos por la cantidad (n) menos uno.

$$\mathbf{V = V(i) / Norma[V(i)] \qquad \lambda = \lambda \ / (n-1)}$$

Ordeno los valores de λ_i de modo que $\lambda_1 > \lambda_2 > .... > \lambda_n$, y de acuerdo a esto también reordeno los eigenvectores V_i. Esto porque la cuantía de un eigenvalor indica la cantidad de información que aporta el componente principal relacionado con este.

Finalmente, se hallan los componentes principales para la matriz (imagen) A

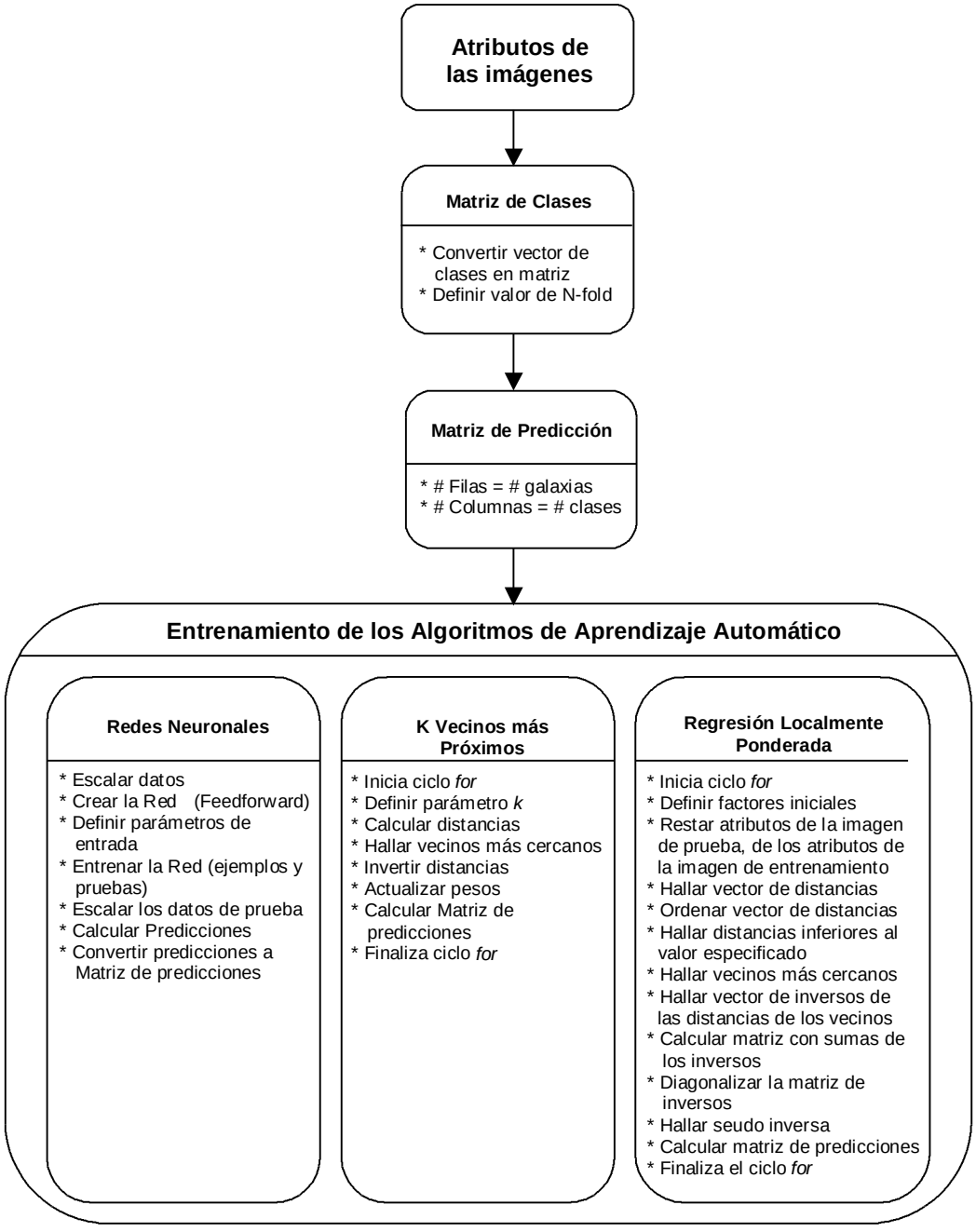
$$\mathbf{CP = V^T * A}$$

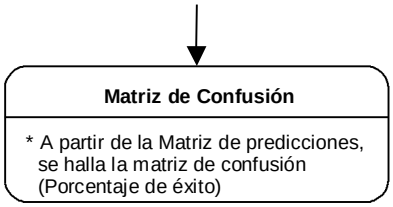
Donde se busca escoger los componentes que brindan más información, desechando los otros, reduciendo así la dimensionalidad de los datos y obteniendo unos buenos descriptores para la imagen.

3.3.3 Aplicación de Algoritmos de Aprendizaje Automático

Estas son las etapas principales en la aplicación de los algoritmos de aprendizaje automático.

Figura 3.19 Etapas de la aplicación de los Algoritmos de Aprendizaje Automático





Para esta etapa se plantearon tres alternativas, tres diferentes algoritmos de aprendizaje automático: las redes neuronales, el algoritmo de los k Vecinos más Próximos (KNN, por sus siglas en inglés K-Nearest Neighbors) y Regresión Localmente Ponderada (LWR, Locally Weighted Regression).

El primer paso es convertir el vector de clases en una matriz que es representativa de los valores de salida.
Se define el factor del algoritmo de validación cruzada (comúnmente es *fold = 10*).
Se define la matriz de predicción que es del mismo tamaño que la matriz representativa de los valores de las clases, es decir, tantas filas como número de galaxias y tantas columnas como número de clases.
Luego, dentro de un ciclo *for* que va desde 1 hasta el factor del algoritmo de validación cruzada:

Se definen tanto el vector de entrenamiento como el vector de testeo, y a continuación se realiza el entrenamiento, dependiendo del algoritmo utilizado:

Para la Red Neuronal:
Se escalan los datos de entrenamiento para ubicarlos en el rango [-1,1], se crea una red tipo feedforward (propagación hacia adelante) , se definen como parámetros de entrada los atributos de las imágenes. Luego se entrena la red neuronal con los parámetros de entrenamiento y de testeo (prueba), se escalan los datos de prueba, se calculan las predicciones y por último se convierten las predicciones a la escala original, dando como resultado la salida de la red neuronal (matriz de predicción).

Para el algoritmo de K Vecinos más Próximos:
Se define el parámetro k (numero de vecinos), se calculan las distancias, se calculan los más cercanos, se invierten las distancias, se actualizan los pesos y se calcula la matriz de predicción, todo esto dentro de un ciclo f*or* que va desde 1 hasta el tamaño del vector de testeo.

Para el algoritmo de Regresión Localmente Ponderada:
Se definen los factores iniciales, como el número de puntos de datos para construir una aproximación local. Se restan los atributos de la imagen para testeo, de los atributos de la imagen para el entrenamiento, luego se halla el

vector de distancias, se ordena este vector, se hallan las distancias que sean inferiores al valor especificado, del más cercano al más lejano; después se hallan los vecinos más cercanos, se hallan el vector con los inversos de las distancias de los vecinos, la matriz que contiene las sumas de dicho vector, luego se diagonaliza esta matriz, se halla la seudo inversa y finalmente se calcula la matriz de predicción. Nuevamente todo lo anterior dentro de un ciclo *for* que va desde 1 hasta el tamaño del vector de testeo.

Acá se cierra el ciclo *for* inicial.

La salida de los tres algoritmos es la matriz de predicción, con la cual se calcula la matriz de confusión, la cual muestra el porcentaje de acierto con respecto a la matriz que representa las clases, es decir, compara las predicciones con las reales.

A continuación se muestra la matriz de confusión obtenida para cada uno de los algoritmos de aprendizaje automático en el mejor caso de acierto:

- *Para Redes Neuronales*

	Esp	**Elip**	**Irr**
Esp	330	10	12
Elip	21	22	0
Irr	13	3	18

Estos resultados corresponden a la siguiente configuración:
Imágenes combinadas, 30 eigenvectores (ACP), red neuronal, fold (factor del algoritmo de validación cruzada) = 10.
Esp = Espirales; Elip = Elípticas; Irr = Irregulares
Los números indican:
330 : Cantidad de galaxias espirales clasificadas correctamente.
10 : Cantidad de galaxias espirales clasificadas como elípticas.
12 : Cantidad de galaxias espirales clasificadas como Irregulares.

21 : Cantidad de galaxias elípticas clasificadas como espirales.
22 : Cantidad de galaxias elípticas clasificadas correctamente.
0 : Cantidad de galaxias elípticas clasificadas como Irregulares.

13 : Cantidad de galaxias irregulares clasificadas como espirales.
3 : Cantidad de galaxias irregulares clasificadas como elípticas.
18 : Cantidad de galaxias irregulares clasificadas correctamente.

En esta corrida se tuvo un porcentaje de acierto de 100*((330+22+18)/429) = 86.25%

- *Para K Vecinos más Próximos*

	Esp	**Elip**	**Irr**
Esp	337	15	0
Elip	18	25	0
Irr	30	1	3

Estos resultados corresponden a la siguiente configuración:
Imágenes combinadas, 30 eigenvectores (ACP), K Vecinos, fold (factor del algoritmo de validación cruzada) = 3.

En esta corrida se tuvo un porcentaje de acierto de 100*((337+25+3)/429) = 85.08%

- *Para Regresión Localmente Ponderada*

	Esp	**Elip**	**Irr**
Esp	331	12	9
Elip	18	25	0
Irr	13	1	20

Estos resultados corresponden a la siguiente configuración:
Imágenes combinadas, 50 eigenvectores (ACP), Regresión Localmente Ponderada, fold (factor del algoritmo de validación cruzada) = 10.

En esta corrida se tuvo un porcentaje de acierto de 100*((331+25+20)/429) = 87.65%

En el capítulo 4 se muestran los resultados completos.

4. RESULTADOS EXPERIMENTALES

La ejecución de los códigos está dividida en tres partes principales:

1. Ejecución de los códigos del preprocesado, es decir, centrado, rotado, recortado y ajustado de las imágenes de las galaxias.
2. Obtención de los valores propios a partir del análisis de componentes principales.
3. Aplicación de los Algoritmos de aprendizaje automático.

En la primera parte, se realizan dos corridas diferentes, en la primera, la entrada es la imagen original y las salida son tres imágenes procesadas de las galaxias, ya centradas, rotadas, recortadas y ajustadas en tamaño estándar (128 x 128 píxeles), cada una representando los componentes r, g y b; en la segunda corrida, el proceso es el mismo excepto que la salida es una sola imagen, pero compuesta por los tres componentes r, g y b (de tamaño 128 x 384 píxeles).

En la segunda parte, a cada imagen obtenida en la etapa anterior se le aplica el proceso de análisis de componentes principales, tomando dos valores diferentes de número de componentes: 30 y 50 teniendo así cuatro diferentes opciones: imágenes separadas (r, g, b) con 30 componentes principales, imágenes separadas con 50 componentes principales, imagen unida (rgb) con 30 componentes principales, e imagen unida con 50 componentes principales.

Finalmente en la última etapa se aplican los tres algoritmos de aprendizaje automático (Redes Neuronales Artificiales (ANN), K-Vecinos más Próximos (KNN) y Regresión Localmente Ponderada (LWR)) a cada una de las opciones que se obtuvieron en la fase anterior, dando como resultado 12 alternativas diferentes:

(r, g, b) + 30 c.p. + ANN
(r, g, b) + 30 c.p. + KNN
(r, g, b) + 30 c.p. + LWR
(r, g, b) + 50 c.p. + ANN
(r, g, b) + 50 c.p. + KNN
(r, g, b) + 50 c.p. + LWR

(rgb) + 30 c.p. + ANN
(rgb) + 30 c.p. + KNN
(rgb) + 30 c.p. + LWR
(rgb) + 50 c.p. + ANN
(rgb) + 50 c.p. + KNN
(rgb) + 50 c.p. + LWR

Para cada una de estas 12 alternativas, se realizaron 50 corridas, tomando como resultado el promedio de las 50 corridas.
En cada corrida se obtuvo una matriz compuesta de tres filas y tres columnas:
Espirales clasificadas como espirales; Espirales clasificadas como elípticas; Espirales clasificadas como Irregulares; (primera fila)
Elípticas clasificadas como espirales; Elípticas clasificadas como elípticas; Elípticas clasificadas como Irregulares; (segunda fila)
Irregulares clasificadas como espirales; Irregulares clasificadas como elípticas; Irregulares clasificadas como Irregulares (tercera fila).

Se utilizó el método de validación N-fold cross validation (Validación cruzada de N iteraciones) con 3, 10 y 20 iteraciones.

En total se realizaron 1800 corridas (2*2*3*50*3) y se obtuvieron los siguientes resultados:

- Para las imágenes separadas, con componentes principales = 30 y con n = 3:

Algoritmo de Aprendizaje	Promedio*
ANN	76,2750583
KNN	80,5081585
LWR	76,4195804

*Porcentaje de galaxias bien clasificadas

- Para las imágenes separadas, con componentes principales = 30 y con n = 10:

Algoritmo de Aprendizaje	Promedio
ANN	76,983683
KNN	80,965035
LWR	74,9137529

- Para las imágenes separadas, con componentes principales = 30 y con n = 20:

Algoritmo de Aprendizaje	Promedio
ANN	77,4219114
KNN	80,9370629
LWR	74,988345

§ Para las imágenes separadas, con componentes principales = 50 y con n = 3:

Algoritmo de Aprendizaje	Promedio
ANN	75,4498834
KNN	80,5454545
LWR	70,0792541

§ Para las imágenes separadas, con componentes principales = 50 y con n = 10:

Algoritmo de Aprendizaje	Promedio
ANN	75,5804196
KNN	80,3403263
LWR	69,995338

§ Para las imágenes separadas, con componentes principales = 50 y con n = 20:

Algoritmo de Aprendizaje	Promedio
ANN	75,7482517
KNN	80,2377622
LWR	70,1025641

§ Para las imágenes unidas, con componentes principales = 30 y con n = 3:

Algoritmo de Aprendizaje	Promedio
ANN	83,4638695
KNN	83,1048951
LWR	85,016317

§ Para las imágenes unidas, con componentes principales = 30 y con n = 10:

Algoritmo de Aprendizaje	Promedio
ANN	84,4382284
KNN	83,6223776
LWR	85,6596737

§ Para las imágenes unidas, con componentes principales = 30 y con n = 20:

Algoritmo de Aprendizaje	Promedio
ANN	84,5501166
KNN	83,8181818
LWR	85,8927739

§ Para las imágenes unidas, con componentes principales = 50 y con n = 3:

Algoritmo de Aprendizaje	Promedio
ANN	82,965035
KNN	83,3659674
LWR	85,039627

§ Para las imágenes unidas, con componentes principales = 50 y con n = 10:

Algoritmo de Aprendizaje	Promedio
ANN	82,965035
KNN	83,3659674
LWR	85,039627

§ Para las imágenes unidas, con componentes principales = 50 y con n = 20:

Algoritmo de Aprendizaje	Promedio
ANN	82,8578089
KNN	83,2307692
LWR	85,5524476

El mejor promedio en todas las corridas correspondió a la combinación:
RGB + LWR + CP=30 + N=20, es decir, Imágenes unidas, con componentes principales = 30, con Algoritmo de aprendizaje = LWR y con N = 20.
Este resultado se utilizará en la etapa del diseño del código en paralelo.

5. PROPUESTA DE PARALELIZACION

En el presente trabajo se presenta una solución al problema de Clasificación de Galaxias, sin embargo, haciendo un análisis tanto en nuestro trabajo como en los otros ya desarrollados en esta área se encuentra una falencia y es que la escala que soportan las aplicaciones planteadas no esta de acuerdo a la realidad que tenemos, donde se debe trabajar con un volumen de datos extremadamente grande, eso teniendo en cuenta que cada vez tenemos mas observadores de nuestro universo y la cantidad de información que manejamos del mismo esta en el orden de 10^8 imágenes por sky survey.

Entonces, como respuesta al problema mencionado se propone la paralelización de algunas funciones de la aplicación, teniendo en cuenta que no todo código es paralelizable y si lo es no todo código paralelo es eficiente.

Un punto a favor dentro de esta propuesta es que estamos trabajando con imágenes que pueden ser vistas de una manera mas básica, como matrices y todas las operaciones efectuadas con estas presentan un alto grado de paralelismo, de acuerdo a esto son las funciones con operaciones matriciales las únicas que se toman en cuenta en el diseño paralelo.

Algoritmo paralelo para el cálculo de los eigenvalores y eigenvectores de una matriz simétrica

Planteamiento Matemático
El algoritmo completo fue desarrollado por Dongarra y Sorensen [1], y esta basado en el algoritmo de iteración QR y la división del dominio (que implica división del trabajo), donde el problema original es partido en dos subproblemas y cada uno de ellos en otros dos subproblemas y así sucesivamente.

- Primero, reducimos la matriz completa A a una matriz tridiagonal T (**A → T)** de acuerdo a la forma de Hessenberg, que es:

$$\mathrm{T} = \begin{bmatrix} a_1 & b_1 & & & \\ b_1 & a_2 & b_2 & & \mathrm{O} \\ & \ddots & \ddots & \ddots & \\ & & b_{n-2} & a_{n-1} & b_{n-1} \\ \mathrm{O} & & & b_{n-1} & a_n \end{bmatrix}$$

[1] J.J. Dongarra & D.C. Soresen. A fully parallel algorithm for the simetric eigenvalue problem. 8:S139-S154, 1987

- Ahora T es dividida en dos matrices simétricas tridiagonales de la mitad del tamaño:

$$\mathrm{T}_1 = \begin{bmatrix} a_1 & b_1 & & & \\ b_1 & a_2 & b_2 & & \mathrm{O} \\ & \ddots & \ddots & \ddots & \\ & & b_{k-2} & a_{k-1} & b_{k-1} \\ \mathrm{O} & & & b_{k-1} & a_k - b_k \end{bmatrix} \quad \mathrm{T}_2 = \begin{bmatrix} (a_{k+1} - b_k) & b_{k+1} & & \\ b_{k+1} & a_{k+2} & b_{k+2} & \mathrm{O} \\ \ddots & \ddots & \ddots & \\ & & & b_{n-1} \\ \mathrm{O} & b_{n-1} & a_n & \end{bmatrix}$$

Tal que:

$$\mathrm{T} = \begin{bmatrix} \mathbf{T}_1 & 0 \\ 0 & \mathbf{T}_2 \end{bmatrix} + b_k \mathbf{z}\mathbf{z}^T$$

Y donde z es:

$$\mathbf{z} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 1 \\ \vdots \\ 0 \end{bmatrix} \begin{matrix} \\ \\ \\ \leftarrow k \\ \leftarrow k+1 \\ \\ \\ \end{matrix}$$

- Después, nosotros podemos diagonalizar T_1 y T_2 usando matrices ortogonales apropiadas Q_1 y Q_2, así: $T_1 = Q_1 D_1 Q^T_1$ y $T_2 = Q_2 D_2 Q^T_2$, donde:

$$\mathbf{D}_1 \equiv \begin{bmatrix} d_1 & & & \\ & d_2 & & \mathrm{O} \\ & & \ddots & \\ \mathrm{O} & & & d_k \end{bmatrix} \quad \mathbf{D}_2 \equiv \begin{bmatrix} d_{k+1} & & & \\ & d_{k+2} & & \mathrm{O} \\ & & \ddots & \\ \mathrm{O} & & & d_n \end{bmatrix}$$

De modo que:

$$\mathbf{D} \equiv \begin{bmatrix} \mathbf{D}_1 & 0 \\ 0 & \mathbf{D}_2 \end{bmatrix}$$

- Basados en la definición anterior podemos introducir ξ y reescribir la matriz T:

$$\xi = \begin{bmatrix} Q_1 & 0 \\ 0 & Q_2 \end{bmatrix}^T z \quad T = \begin{bmatrix} Q_1 & 0 \\ 0 & Q_2 \end{bmatrix} \left\{ \begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix} + b_k \xi \xi^T \right\} \begin{bmatrix} Q_1 & 0 \\ 0 & Q_2 \end{bmatrix}^T$$

,

Donde podemos observar que la matriz T es similar a la matriz G definida:

$$G \equiv D + b_k \, \xi \, \xi^T$$

- Y entonces encontramos que los valores λ_i de G están dados por la ecuación:

$$1 + b_k \sum_{i=1}^{n} \frac{\xi_i^2}{d_i - \lambda} = 0.$$

y el problema se reduce a hallar las raíces de la función:

$$f(\lambda) = 1 + b_k \sum_{i=1}^{n} \frac{\xi_i^2}{d_i - \lambda}.$$

- Ahora, teniendo los eigenvalores λ_i de G que también son los eigenvalores de A, podemos calcular los eigenvectores. Primero, los correspondientes eigenvectores de G dados por:

$$Y_i = (D - \lambda_i I)^{-1} \xi$$

- Y finalmente encontrar los eigenvectores de la matriz original A dados por:

$$v_i = \begin{bmatrix} Q_1 & 0 \\ 0 & Q_2 \end{bmatrix} y_i.$$

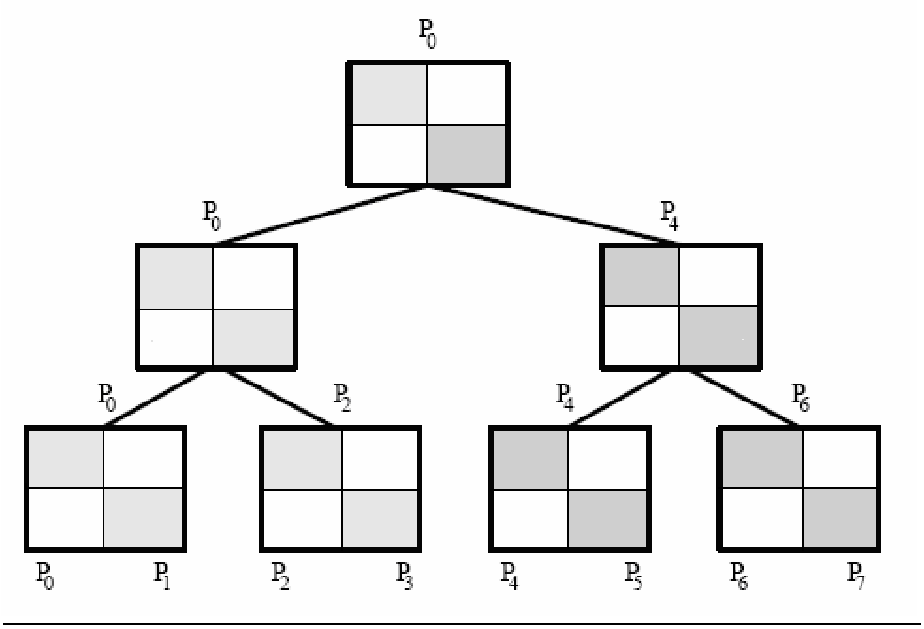
Esquema del Algoritmo

El nodo maestro toma la matriz original A y la transforma a la matriz tridiagonal T mediante el algoritmo de Hessenberg.

Después toma la matriz T y la divide en 2 submatrices, cosa que hace sucesivamente hasta tener tantas submatrices como nodos, asignando una porción de datos y trabajo a cada nodo.

$$T = \begin{bmatrix} T_1 & 0 \\ 0 & T_2 \end{bmatrix}$$

Figura 5.1 División de dominio para la matriz Tridiagonal T



fuente: KARNIADAKIS, George y KIRBY, Robert. Parallel Scientific Computing in C++ and MPI.

Después, cada nodo toma su parte de los datos y hace los cálculos respectivos para hallar D, Q, b_k, y ξ (locales) sobre el dominio que le corresponde.

$T_1 = Q_1 D_1 Q^T_1$ y $T_2 = Q_2 D_2 Q^T_2$

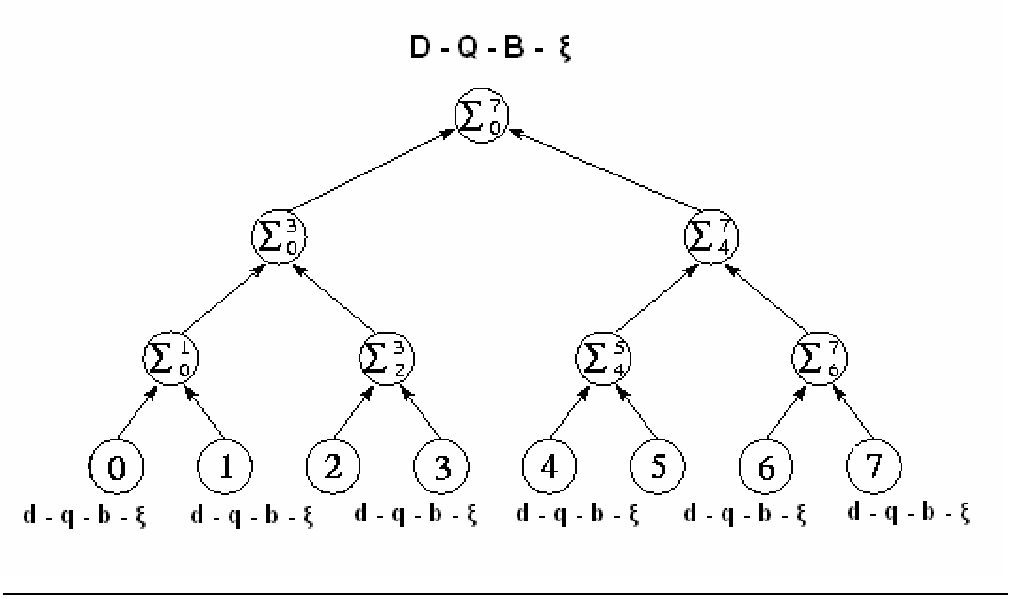
$$\mathbf{D} \equiv \begin{bmatrix} \mathbf{D}_1 & 0 \\ 0 & \mathbf{D}_2 \end{bmatrix}$$

$$\xi = \begin{bmatrix} \mathbf{Q}_1 & 0 \\ 0 & \mathbf{Q}_2 \end{bmatrix}^T \mathbf{z}$$

$$b_k$$

Seguidamente el nodo maestro reúne los resultados del los cálculos generados por cada uno de los nodos esclavos y con esto forma las soluciones globales para la matriz global D, Q, b_k, y ξ.

Figura 5.2 Reducción Maestro – Esclavo en forma de árbol



fuente: KARNIADAKIS, George y KIRBY, Robert. Parallel Scientific Computing in C++ and MPI.

Después de que el maestro reúne las soluciones locales y arma la solución global D, Q, b_k, y ξ plantea la matriz $G \equiv D + b_k\ \xi\ \xi^T$, calcula sus eigenvectores y eigenvalores.

Finalmente se aplica una transformación a los eigenvectores para G para que se conviertan en los eigenvectores de A, entonces:

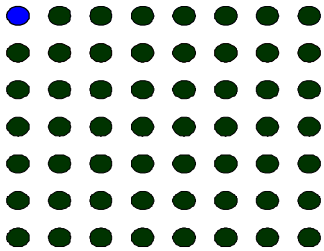
$$f(\lambda) = 1 + b_k \sum_{i=1}^{n} \frac{\xi_i^2}{d_i - \lambda}.$$

$$Y_i = (D - \lambda_i I)^{-1} \xi$$

$$\mathbf{v}_i = \begin{bmatrix} \mathbf{Q}_1 & 0 \\ 0 & \mathbf{Q}_2 \end{bmatrix} \mathbf{y}_i.$$

Diseño Paralelo:

- Partición
  La forma en que se trabaja una imagen es a través de una matriz, donde cada elemento es un píxel que contiene la intensidad luminosa para ese punto. Esta característica hace que la descomposición de dominio mas agresiva sea la de un punto o píxel por cada tarea.

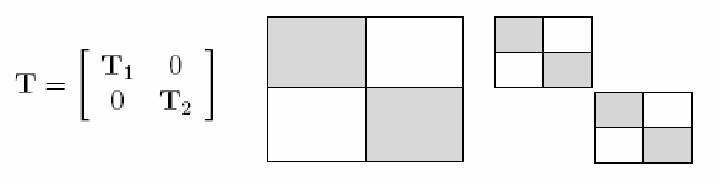


  Cada tarea mostrada en la figura mantiene actualizado un punto de la matriz, y es responsable el cómputo relacionado con el. La cantidad de tareas que se crean es m*n, donde m es la cantidad de filas y n la cantidad de columnas.

- Comunicación
  El requerimiento de comunicación es global y se da después de que el maestro calcula la matriz tridiagonal, aquí el envía la parte correspondiente a cada uno de los nodos esclavos para que haga el desarrollo y calculo de su subproblema.
  La siguiente comunicación se da cuando todos los nodos esclavos envían el calculo realizado al nodo maestro para que el lo reciba y organice de modo que genere la solución completa y general del problema.

- Agrupación
  Para este problema específico nos exige una agrupación por cuadrantes, ya que esa es la forma de la matriz que vamos a solucionar y dividir

Figura 5.3 División de Dominio para la matriz T

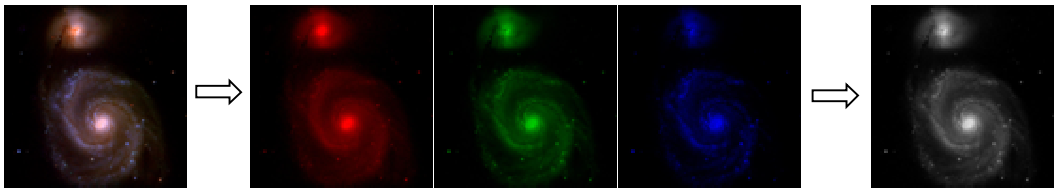


- Asignación
  La asignación debe ser cuidadosa, pues si observamos detenidamente, la comunicación se desarrolla cuando un nodo de un nivel superior se comunica con otros dos y así seguidamente, dicho envío en árbol exige que la cantidad de nodos disponibles sea múltiplo de las potencias de 2, es decir 2^2=4, 2^3=8, 2^4=16 etc…

Algoritmo paralelo para el generar una imagen promediada a partir de una imagen en color

Planteamiento:
Esta función toma una imagen a color y hace un promedio píxel a píxel sobre cada una de las bandas (r,g,b) generando una imagen en escala de grises (una sola banda).

Figura 5.4 Proceso de Promediado

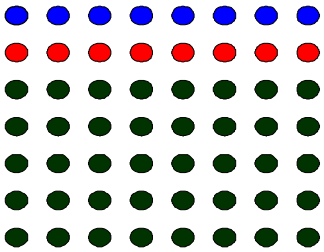


```
para fila=1:numero_filas haga
   para columna=1:numero_columnas haga
      para banda=1:numero_bandas haga
        imagen_promediada = imagen_original(fila, columna, banda) /
numero_bandas
      fin
  fin
fin
```

Justificación:
Cuando se habla de imágenes astronómicas se puede pensar inmediatamente en una alta resolución, lo que involucra un gran nivel de detalle, peso del archivo y cantidad de píxeles.
El trabajo con imágenes de este tipo tiene un costo de tiempo que aumenta a medida que crece la imagen, sin embargo, el algoritmo presentado se puede hacer una buena división del dominio incurriendo en un mínimo de comunicación.

Diseño Paralelo:

- Partición
  La forma en que se trabaja una imagen es a través de una matriz, donde cada elemento es un píxel que contiene la intensidad luminosa para ese punto. Esta característica hace que la descomposición de dominio sea la más adecuada para aplicar en nuestro problema.
  Tratando de buscar la mayor concurrencia posible, se ha escogido inicialmente realizar la descomposición más agresiva, que en este caso es definir una tarea por cada fila de la matriz.

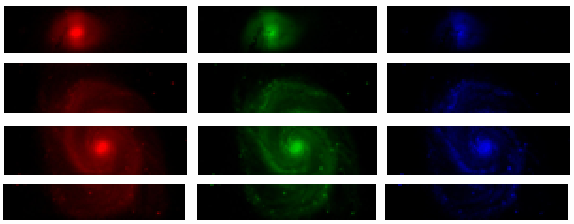


  Cada tarea mostrada en la figura mantiene actualizada la fila de la matriz, y es responsable el cómputo relacionado con ella. La cantidad de tareas que se crean es m, donde m es la cantidad de filas y n la cantidad de columnas.

- Comunicación
  El requerimiento de comunicación es global y se da al inicio y al final, al inicio pues "todas" las tareas deben tener la información sobre los puntos sobre los cuales es responsable del cómputo, entonces el nodo maestro debe hacer una partición del dominio y comunicar a cada esclavo la parte de la imagen con la cual debe trabajar (parte de R, parte de G y parte de B).
  Además, al final el nodo maestro debe recoger las soluciones que serán los promedios realizados por cada uno de los nodos.

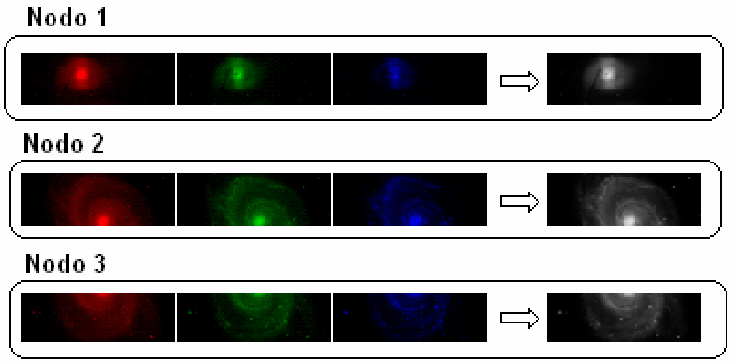
- Agrupación
  Con descomposición realizada en la primera etapa se crearían m tareas, por ejemplo para N Pixeles en X=1200 y N Pixeles en Z=800 el número de tareas ascendería a 1200. Esta cantidad de tareas es mucho mayor de la necesitada para conservar la flexibilidad, entonces es preferible agrupar varias tareas (filas) de modo que haya una distribución equitativa que lleve a un mayor tiempo de trabajo y una menor comunicación.

Figura 5.5 División del dominio en el proceso de promediado



- Asignación
  De acuerdo a la agrupación hecha se asignan las tareas a los nodos disponibles los cuales arrojaran sus respuestas locales

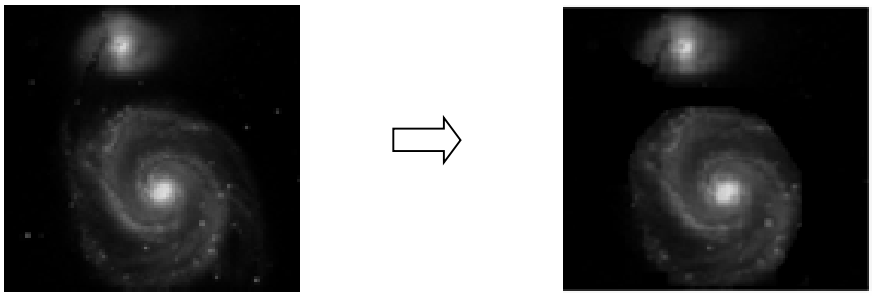
Figura 5.6 Asignación de trabajo en el proceso de promediado



Algoritmo paralelo para el filtrado de una imagen

Planteamiento:
Esta función toma una imagen y la recorre haciendo un test píxel a píxel donde si la intensidad sobre ese píxel supera un umbral definido este permanece igual, pero si no se le asigna un valor de cero (color negro)

Figura 5.7 Filtrado de una imagen

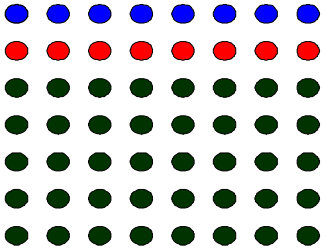


```
para fila=1:numero_filas haga
   para columna=1:numero_columnas haga
      si imagen(fila, columna) < umbral
         entonces
            imagen(fila, columna) = 0
         sino
              imagen(fila, columna) = imagen(fila, columna)
      fin si
  fin para
fin para
```

Justificación:
El procedimiento de filtrado en una imagen es usado frecuentemente, y cuando hablamos de una gran cantidad de imágenes o de imágenes de gran tamaño este se hace muy pesado, es por eso que se trata de dar una solución paralela para mejorar el rendimiento.

Diseño Paralelo:

- Partición
  Tratando de buscar la mayor concurrencia posible, se ha escogido inicialmente realizar la descomposición más agresiva, que en este caso es definir una tarea por cada fila de la matriz.

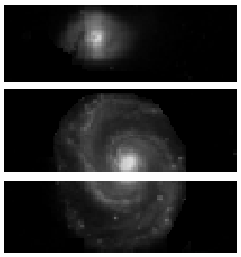


  Cada tarea mostrada en la figura mantiene actualizada la fila de la matriz, y es responsable el cómputo relacionado con ella. La cantidad de tareas que se crean es m, donde m es la cantidad de filas y n la cantidad de columnas.

- Comunicación
  El requerimiento de comunicación es global y se da al inicio y al final, al inicio pues "todas" las tareas deben tener la información sobre los puntos sobre los cuales es responsable del cómputo, entonces el nodo maestro debe hacer una partición del dominio y comunicar a cada esclavo la parte de la imagen con la cual debe trabajar.
  Además, al final el nodo maestro debe recoger las soluciones que serán las partes umbralizadas para cada uno de los nodos.

- Agrupación
  Con la descomposición realizada en la primera etapa se crearían m tareas, por ejemplo para N Pixeles en X=1200 y N Pixeles en Z=800 el número de tareas ascendería a 1200. Esta cantidad de tareas es mucho mayor de la necesitada para conservar la flexibilidad, entonces es preferible agrupar varias tareas (filas) de modo que haya una distribución equitativa que lleve a un mayor tiempo de trabajo y una menor comunicación.

Figura 5.8 Agrupación dentro del proceso de filtrado



- Asignación
  De acuerdo a la agrupación hecha se asignan las tareas a los nodos disponibles los cuales arrojaran sus respuestas locales (partes filtradas).

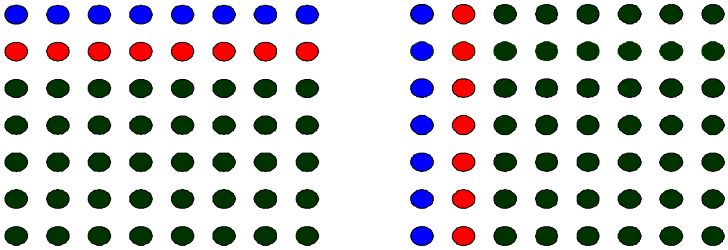
Algoritmo paralelo para multiplicacion de matrices

Planteamiento:
Dentro del trabajo con imágenes la multiplicación de matrices es una labor bastante común, y dado que hay tanta recurrencia hacia esta función serian enormemente considerables las mejoras obtenidas generalmente si se desarrolla la paralelización de esta sencilla función.

```
para i = 1: filas_A haga
   para j = 1:columnas_B haga
        para k = 1:columnas_A haga
              c(i,j) = a(i,k) * b(k,j)
        fin para
  fin para
fin para
```

Diseño Paralelo:

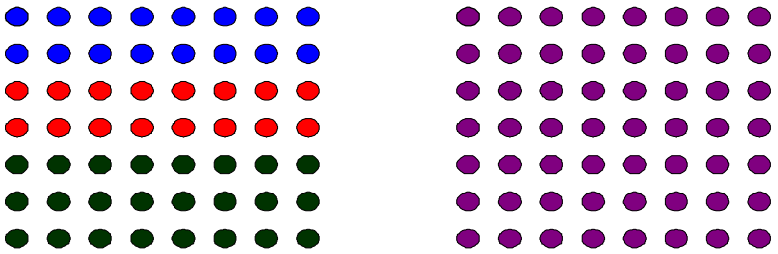
- Partición
  Tratando de buscar la mayor concurrencia posible, se ha escogido inicialmente realizar la descomposición más agresiva, que en este caso es definir una tarea por cada fila de la matriz A y una tarea por cada columna de la matriz B.



Cada tarea mostrada en la figura mantiene actualizada la fila o columna de la matriz que le corresponda, es responsable el cómputo relacionado con ella. La cantidad de tareas que se crean es m+n, donde m es la cantidad de filas y n la cantidad de columnas.

- Comunicación
  La comunicación se da dos veces, al inicio cuando el maestro hace la división del dominio y envía a cada nodo el trabajo que tiene que realizar y al final cuando cada trabajador envía al maestro la parte que calculo de la matriz resultante.

- Agrupación
  Una forma eficiente de agrupación para la multiplicación de matrices es hacer una partición por filas en la matriz A, y hacer una replica de la matriz B pues esto facilita el cálculo y el ordenamiento de la matriz respuesta.



- Asignación
  De acuerdo a la agrupación hecha se asignan las tareas a los nodos disponibles, teniendo en cuenta el número de filas que por nodo debe haber: total_filas / total_nodos, si la division es entera; pero si existe un residuo este debe repartirse de manera equitativa entre los nodos disponibles.
  La matriz B debe ser replicada completamente en todos los nodos.

6. CONCLUSIONES

- El proceso de estandarización de las imágenes permite minimizar las variaciones en los objetos a clasificar y de esta forma obtener mejores resultados.

- Debido a que las imágenes adquiridas no están en las condiciones ideales para su procesamiento, el tratamiento digital de imágenes se presenta como una buena técnica para ajustar las imágenes de acuerdo a las necesidades que se presentan.

- El análisis de componentes principales facilita una primera interpretación sobre los ejes de variabilidad de la imagen, y nos permite identificar aquellos rasgos que aparecen en la mayoría de las bandas.

- ACP es una excelente técnica para el uso previo a la clasificación ya que brinda unos parámetros descriptores adecuados y propios del objeto a clasificar.

- Dentro de los algoritmos de aprendizaje automático, los algoritmos por distancias brindan una mejor aproximación a la correcta clasificación, obteniendo mejores resultados que las Redes Neuronales (Para este trabajo).

- Ya que las imágenes con las que se trabaja son matrices y la mayor parte de los procesos que se realizan son operaciones matriciales, un alto porcentaje del código se puede paralelizar.

- La combinación de técnicas computacionales como el tratamiento digital de imágenes, el análisis de componentes principales y los algoritmos de aprendizaje automático, brindan buenos resultados dentro del proceso de clasificación de galaxias.

7. RECOMENDACIONES

Para trabajos futuros que estén relacionados:

- Se recomienda trabajar con imágenes en formatos que suministren más información acerca de la galaxia. Las imágenes con formato ".fit" son las estándares para astronomía y tienen muy completa información acerca de la imagen como tal y también del sector de la bóveda celeste donde se capturó la imagen, el tamaño (en segundos de arco) de la galaxia, etc..

- La mayoría de imágenes con las que se trabajaron tienen una ubicación frontal, lo cual facilita el proceso. Se recomienda tener en cuenta imágenes que estén en condiciones más complicadas y que requieran de algoritmos para deproyectarlas.

- Se trabajó con las bandas R, G, y B de las imágenes, pues brindan mayor información que la imagen compuesta. Se recomienda adquirir imágenes en otras bandas del espectro electromagnético (I, H/J/K, y B), pues así se tendría un rango de acción más amplio, brindando información de otras características de la galaxia, no apreciables en el rango visible.

- Se demostró que el Análisis de componentes principales es una herramienta computacional muy buena para hallar las características de una imagen de galaxia. Sin embargo, combinando el ACP con otros descriptores, especialmente morfológicos, se deberá tener un muy buen caso de estudio para la mejora de los resultados.

- Se recomienda utilizar el algoritmo de Support Vector Machine, como algoritmo de aprendizaje automático así como otros tipos diferentes de Redes Neuronales, etc.

- Por último, se recomienda llevar a cabo la implementación del algoritmo paralelo.

BIBLIOGRAFÍA

CAPITULO 2. FUNDAMENTACION TEORICA

PROTHEROE, W. M.; CAPRIOTTI, E. R.; NEWSOM, G. H. Exploring the Universe. 2ª Edición. Editorial Charles E. Merrill Publishing Company. Ohio State University. USA. 1986.

BAKULIN, P.I., KONONOVICH, E.V., MOROZ, V.I. Curso de Astronomía General. Editorial MIR. Moscú, 1983.

SNOW, Theodore P. Essentials of the Dynamic Universe: An Introduction to Astronomy. Ed. West Publishing Co., St. Paul, USA. 1984.

http://www.astromia.com/universo/clasegalaxias.htm

GONZÁLEZ, Rafael C.; WOODS, Richard E. Tratamiento Digital de Imágenes. Ed. Addison-Wesley / Diaz de Santos. Washington, USA. 1996.

PAJARES MARTINSANZ, Gonzalo; DE LA CRUZ GARCIA, Jesús M. Visión por Computador: Imágenes Digitales y Aplicaciones. Ed. Alfaomega / RAMA. México / Madrid. 2002

RUSS, John C. The Image Processing Handbook. Ed. CRC Press. Boca Ratón, USA, 1999.

MARTINEZ, Víctor E; MENDOZA CASTELLANOS Alfonso. Implementación de un Modelo Computacional para Clasificación Normal - Displásica de las Células Escamosas de Citologías Cérvico Uterinas. Tesis de Grado. UIS, Bucaramanga, Colombia. 2004

HALL, Ernest L. Computer Image Processing and Recognition. Academic Press. 1979.

BERRY, R. H. ; HOBSON, M. P. ; WITHINGTON, S. Modal Decomposition of Astronomical Images with Application to Shapelets. Astrophysics Group, Cambridge, Inglaterra. 2002

REFREGIER, Alexander. Shapelets I : A Method for Image Analysis. Institute of Astronomy. Cambridge, Inglaterra. http://www.astro.caltech.edu/~rjm/shapelets/

CLAVIJO MENDEZ, Jairo Alfonso. Análisis de Componentes Principales – ACP. Universidad del Tolima. Ibagué, Colombia

FERRERO, Susana Beatriz. Análisis de Componentes Principales en Teledetección. Consideraciones estadísticas para optimizar su interpretación. Universidad Nacional de Río Cuarto. Córdoba, Argentina.

GUZMÁN DE LEON, Alejandro. Procesamiento Digital de Imágenes Colposcópicas. UAM, Iztapalapa, México

MITCHELL, Tom M. Machine Learning. Ed. McGraw-Hill Science/Engineering/Math. Pittsburgh, USA. 1997

KULKARNI, Arun D. Computer Vision and Fuzzy - Neural Systems. Prentice Hall. 2001

HILERA GONZALEZ, José Ramón. Redes Neuronales Artificiales: Fundamentos, Modelos y Aplicaciones. Ed. Alfaomega / RAMA. Bogota / Madrid 2002.

http://es.wikipedia.org/wiki/Aprendizaje_Automático

SALINAS, Renato. Red Neuronal de Arquitectura Paramétrica en Reconocimiento de Rostros. Universidad de Santiago de Chile. http://cabierta.uchile.cl/revista/17/articulos/paper4/

http://www-etsi2.ugr.es/depar/ccia/rf/www/tema3_00-01_www/tema3_00-01_www.html

FOSTER, Ian. Designing and Building Parallel Programs, Ed. Addison-Wesley. University of Chicago. 1995.

KARNIADAKIS, George y KIRBY, Robert. Parallel Scientific Computing in C++ and MPI. Ed. Cambridge University. 2001.

BERTSEKAS, Dimitri. Parallel And Distributed Computation: Numerical Methods. Prentice-Hall. Michigan Institute of Technology. Boston, USA. 1989.

SUK, Minsoo y BHANDARKAR, Suchendra. Three - Dimensional Object Recognition from Range Images. Ed. Springer - Verlag. Tokyo. 1992

DOWN, Kevin y SEVERANCE, Charles. High Performance Computing. Ed. O´really. 1998

CAPITULO 3. IDENTIFICACIÓN DE GALAXIAS: DESARROLLO SECUENCIAL

PRESSMAN, Roger. Ingeniería del Software: Un Enfoque Práctico. Ed. Mc Graw Hill. 2001.

VERA, Nelson; DOTTORI, Horacio; PUERARI, Ivanio. Estudio Dinámico y Morfológico de Galaxias Espirales por Medio de la Transformada de Fourier. Instituto Nacional de Astrofísica Óptica y Electrónica, Puebla, México. Universidade Federal do Rio Grande do Sul, Brasil.

DE LA CALLEJA, Jorge; FUENTES, Olac. Machine learning and image análisis for morphological galaxy classification. Instituto Nacional de Astrofísica Óptica y Electrónica. Puebla, México. 2003

BALL, N. M.; LOVEDAY, J. et al. Galaxy types in the Sloan Digital Sky Survey using supervised artificial neural networks. Astronomy Centre, University of Sussex, Brighton, Inglaterra. 2003

YAMAUCHI, Chisato; ICHIKAWA, Shin-Ichi et al. Morphological classification of galaxies using photometric parameters: the concentration index versus the coarseness parameter. National Astronomical Observatory. Tokyo, Japón. 2005.

STORRIE-LOMBARDI, M. C.; LAHAV O. et al. Morphological classification of galaxies by Artificial Neural Networks. Institute of Astronomy. Cambridge, Inglaterra. 1992

CAPITULO 5. PROPUESTA DE PARALELIZACION

FOSTER, Ian. Designing and Building Parallel Programs, Ed. Addison-Wesley. University of Chicago. 1995.

BERTSEKAS, Dimitri. Parallel And Distributed Computation: Numerical Methods. Prentice-Hall. Michigan Institute of Technology. Boston, USA. 1989.

DONGARRA, J. J.; SORENSEN, D. C. A fully parallel algorithm for the simetric eigenvalue problem. 1987