Antonio Pérez Carrasco
J. Ángel Velázquez Iturbide
Francisco J. Almeida Martínez

Revisión Bibliográfica de la Representación de Problemas de la Técnica “Divide y Vencerás”

Número 2012-02

Serie de Informes Técnicos DLSI1-URJC
ISSN 1988-8074
Departamento de Lenguajes y Sistemas Informáticos I
Universidad Rey Juan Carlos

Índice

Índice de ilustraciones

Revisión Bibliográfica de la Representación de Problemas de la Técnica "Divide y Vencerás"

Antonio Pérez Carrasco[1], J. Ángel Velázquez Iturbide[1], Francisco J. Almeida Martínez[1]

[1] LITE – Laboratory of Information Technologies in Education
Departamento de Lenguajes y Sistemas Informáticos I
Escuela Técnica Superior de Ingeniería Informática
Universidad Rey Juan Carlos, Madrid, España
{ antonio.perez.carrasco,angel.velazquez,francisco.almeida}@urjc.es

Abstract. Se presenta la revisión bibliográfica realizada con el fin de conocer qué tipos de representaciones son las más comunes para ilustrar algoritmos de la técnica "divide y vencerás". Se establece una clasificación con distintos tipos de problemas con el fin de manejar más fácilmente el catálogo completo de problemas encontrados, ayudando a entender las diferentes tipologías de problemas y permitiendo aportar visualizaciones adaptadas a tales clases.

Keywords: Visualización, Algoritmo, Divide y vencerás, Revisión bibliográfica.

1. Introducción

La diversa bibliografía que existe dedicada a la docencia de la algoritmia emplea representaciones gráficas para complementar e ilustrar las explicaciones textuales que desarrollan los contenidos. Tales ilustraciones permiten introducir ejemplos que ilustren conceptos de manera general o que tengan un carácter más específico sobre los algoritmos que se desarrollan en las obras bibliográficas.

El catálogo existente de algoritmos diseñados bajo la técnica "divide y vencerás" es muy amplio, si bien se pueden establecer diversas clasificaciones atendiendo a múltiples parámetros que permiten agruparlos en función de ciertas características de los propios problemas. Se repasa a continuación el procedimiento llevado a cabo y los resultados obtenidos, en forma de clasificación de problemas, que son repasados para aportar si cuentan o no con representación gráfica en la bibliografía consultada.

En el apartado 2 se revisa la metodología empleada, en el apartado 3 se presenta la clasificación realizada, mientras que en los apartados siguientes se repasan las distintas clases de problemas, aportando para ellos las representaciones encontradas. Tras ello, aparecen las conclusiones que cierran este trabajo.

2. Metodología Empleada

El trabajo que aquí se presenta comenzó desde la hipótesis de que las representaciones empleadas en los libros de referencia son las más extendidas, aceptadas y útiles en el contexto docente. Por ello es por lo que se decidió iniciar una exploración de la bibliografía existente sobre algoritmia disponible en la biblioteca universitaria, seleccionando sólo aquellas obras centradas en el diseño de los algoritmos y que cuentan con capítulos o apartados enfocados en la técnica de diseño “divide y vencerás”.

Los contenidos que se pretendía recolectar eran, concretamente, cualquier representación gráfica, independientemente del tamaño o de sus propiedades específicas, que fuese utilizada para ilustrar, bien en un sentido genérico o con un ejemplo o caso concreto, un concepto o algoritmo, bien de manera parcial o total. Por tanto, las representaciones tales como los diagramas, las secuencias cronológicas de estados, elementos esquemáticos o incluso representaciones especiales, fueron los objetivos de esta búsqueda realizada.

La búsqueda bibliográfica fue realizada tomando como base documental un total de 15 obras [1][2][3][4][5][6][7][8][9][10][11][12][13][14][15], recopilando información sobre las características de cada representación y estableciendo una relación entre cada problema y qué representaciones tenía disponibles una vez fue revisada toda la bibliografía.

Así, se alcanzó un conjunto de 20 problemas diferentes diseñados bajo la técnica “divide y vencerás”. Todos los problemas encontrados fueron catalogados, incluso si no tenían ninguna representación que los ilustrara. Con esto se consiguió comenzar a distinguir tipos de problemas que no solían admitir representaciones gráficas de otros que ofrecían un conjunto variado.

3. Clasificación de Algoritmos

La clasificación de los algoritmos encontrados abre la posibilidad de analizar más ágilmente las carencias y propiedades que pueden encontrarse en las representaciones de ciertos grupos de problemas o los factores que permiten generar más fácilmente visualizaciones genéricas expresivas para ciertos algoritmos. Esta clasificación se ha realizado tomando criterios estructurales de los problemas objeto de estudio, lo que ha permitido desarrollar la clasificación que se expone a continuación:

1) Algoritmos que resuelven problemas matemáticos, que pueden manejar o no estructuras de datos. Son algoritmos para los que en general resulta complicado ofrecer una representación gráfica pero en algunos casos sí logran encontrarse visualizaciones que ayuden a comprender su funcionamiento. Algunos ejemplos de este tipo de problemas son la multiplicación de matrices y el cálculo del factorial.
2) Algoritmos que utilizan estructuras de datos (tales como vectores o matrices) devolviendo un valor único de retorno (por ejemplo, un algoritmo de búsqueda opera sobre un vector o matriz y devuelve un valor numérico que indica la posición o un valor inválido si éste no se encuentra).

3) Algoritmos que emplean estructuras de datos (vectores, matrices), realizando modificaciones sobre tales estructuras, siendo ése el propio resultado del algoritmo (por ejemplo, algoritmo de ordenación, donde el resultado queda en la propia estructura manejada), no devuelven ningún valor específico.
4) Algoritmos de carácter geométrico, tratando puntos en el espacio y haciendo uso de sus propiedades matemáticas para realizar diferentes cálculos sobre tales elementos.
5) Existe un quinto grupo que aglutina los restantes algoritmos no clasificados en los grupos anteriores.

Ésta es una clasificación parecida en algunos aspectos a la que propuso Stern, que está basada en la gestión realizada de las estructuras de datos que manejan los problemas, lo que permitió definir tres categorías diferenciadas:

1) Recorrido por una estructura (por ejemplo, algoritmo de búsqueda).
2) Manipulación de elementos de una estructura (por ejemplo, algoritmo de ordenación).
3) Construcción de datos en una estructura (por ejemplo, algoritmo de inserción).

Así, las categorías 2 y 3 podrían corresponderse en gran medida con la categoría 3 de la clasificación propuesta, mientras que la categoría 1 de la catalogación de Stern se asemeja a la categoría 2 de nuestra propuesta.

Una vez presentada la clasificación realizada de los problemas, se revisan sus cinco categorías, exponiendo y analizando las visualizaciones disponibles en los libros. Se podrá observar que los problemas de las distintas categorías cuentan con representaciones muy similares en la bibliografía en ciertos casos.

4. Problemas Matemáticos

Estos problemas dan una solución a problemas relacionados estrechamente con cuestiones teóricas y de carácter abstracto que provienen del campo matemático. Algunos ejemplos de este tipo de problemas son:

1) La multiplicación de enteros de gran tamaño.
2) La multiplicación de matrices con algoritmos avanzados más eficientes.
3) La exponenciación.
4) El cálculo de factoriales.
5) El cálculo de la serie de números de Fibonacci.
6) Otros más complejos como la convolución de funciones y la transformada de Fourier.

Todos estos problemas pueden ser resueltos aplicando la técnica “divide y vencerás” pero aún no ha sido encontrada una representación gráfica verdaderamente útil de algunos de estos problemas, por lo que la exposición que se puede encontrar en la bibliografía suele limitarse al desarrollo textual y las expresiones matemáticas. Es éste el caso de la multiplicación de números grandes [2][3][4][10], la multiplicación de matrices en todas sus variantes [2][3][4][9][13][14][15], del cálculo de convoluciones y del de transformadas de Fourier [10][11].

En ocasiones algunos autores hacen uso de pequeños diagramas que intentan situar al lector en el objetivo del algoritmo o explicar algunas propiedades. Algunos ejemplos son la representación estrictamente matemática de la multiplicación de matrices [9][13][14], o la representación de la multiplicación de números, escribiendo uno debajo del otro y dejando huecos para el algoritmo iterativo clásico que se aplica en el cálculo manual [10] y haciendo sobre ellos una primera división en dos mitades que sirva para introducir el algoritmo [3].

Sí existen, no obstante, ciertos problemas matemáticos que sí que cuentan en la bibliografía con una representación gráfica que los ilustre, como son el cálculo del factorial, la exponenciación y el cálculo de la serie de los números de Fibonacci. De ellos sólo el cálculo del factorial cuenta con representación gráfica en los libros consultados [6] para mostrar el comportamiento de la pila del programa, mientras que para la exponenciación [3][4] y el cálculo de la serie de los números de Fibonacci [3] los autores sólo se apoyan en explicaciones textuales y expresiones matemáticas, si bien el árbol de activación es una visualización que puede ofrecer unos resultados ampliamente satisfactorios.

Por su parte, el problema de multiplicación de matrices no puede admitir una representación genérica fácil de diseñar ya que existen numerosos algoritmos que cumplen esta labor ofreciendo unos procedimientos diferentes cada uno de ellos.

4.1 Multiplicación de Enteros de Gran Tamaño

Para este problema se han encontrado las siguientes representaciones en [2][3][4][10][14][15]:

- Explicación textual.
- Esquema de la división de los números
- Representación matemática (esquema de escritura para cálculo manual)

En la Ilustración 1 y en la Ilustración 2 se muestra la representación esquemática de los dos números, representados por cajas que son divididas en dos mitades. Cada mitad es renombrada y su nombre es empleado en expresiones matemáticas. Se presentan las imágenes extraídas de [3] y [2] respectivamente. También se ofrece en la Ilustración 3 la representación clásica del algoritmo, utilizada ocasionalmente en la bibliografía, obtenida desde [10].

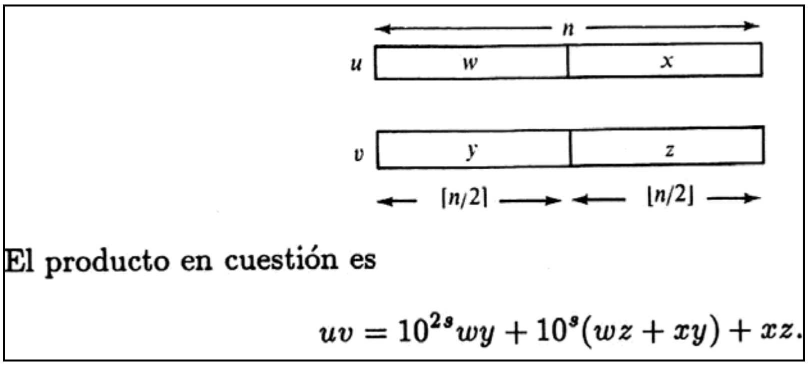


Ilustración 1, esquema de la multiplicación de enteros

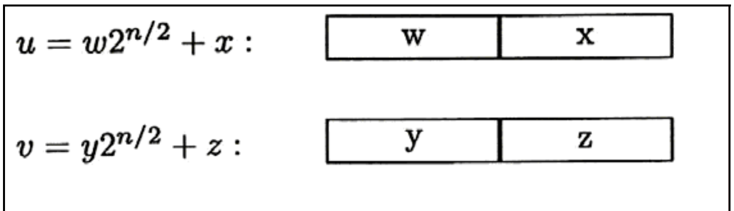


Fig. 6.6 Multiplication of two large integers.

Ilustración 2, esquema de la multiplicación de números

$$
\begin{array}{r}
12\\
\times\,13\\
\hline
36\\
12\\
\hline
156
\end{array}
\qquad\qquad
\begin{array}{r}
1100\\
\times\,1101\\
\hline
1100\\
0000\\
1100\\
1100\\
\hline
10011100
\end{array}
$$

(a) (b)

Ilustración 3, escritura clásica de la multiplicación de números

4.2 Multiplicación de Matrices

Para la multiplicación de matrices sólo se ha dado con la explicación meramente textual y un tipo de distribución gráfica a pesar de aparecer en varias obras [2][3][4][9][13][14][15]:

- Explicación textual.
- Disposición matemática.

Tal y como se ha explicado, resulta muy complejo para los autores encontrar una representación que resulte realmente ilustrativa sobre el proceso que realizan diversos algoritmos avanzados, de ahí que las exposiciones de este problema se basen exclusivamente en la explicación textual o, en algunos casos, apoyado en las propias expresiones matemáticas que solucionan este problema de multiplicación. La disposición matemática que aparece en la Ilustración 4 se ha obtenido desde [14]:

$$
\begin{bmatrix} c_{00} & c_{01} \\ c_{10} & c_{11} \end{bmatrix} = \begin{bmatrix} a_{00} & a_{01} \\ b_{10} & b_{11} \end{bmatrix} * \begin{bmatrix} b_{00} & b_{01} \\ b_{10} & b_{11} \end{bmatrix}
$$
$$
= \begin{bmatrix} m_1 + m_4 - m_5 + m_7 & m_3 + m_5 \\ m_2 + m_4 & m_1 + m_3 - m_2 + m_6 \end{bmatrix}
$$

Ilustración 4, multiplicación de matrices

4.3 Exponenciación

Para el cálculo de la exponenciación, explicado en [3][4], sólo se ha encontrado un tipo de exposición:

- Explicación textual.

A pesar de que ciertas representaciones, como son el árbol de activación o la pila de control, pueden ser útiles para la ilustración del problema, ninguna de las obras consultadas aporta ningún tipo de ilustración para un problema que, por otra parte, suele ser bien conocido por los estudiantes y además resulta fácil de aprender al no requerir de ninguna estructura de datos que manejar y basarse únicamente en el decremento de un número para realizar las sucesivas llamadas recursivas.

4.4 Cálculo de Factoriales

Para el cálculo del número factorial sólo se ha encontrado el siguiente tipo de representación en [6]:

- Pila de control.

Tan sólo se encontró una representación gráfica para ilustrar el problema del cálculo del factorial, se muestra a continuación en la Ilustración 5 tal representación extraída de [6], donde se puede apreciar, más que una representación, una secuencia de representaciones de la citada pila de control.

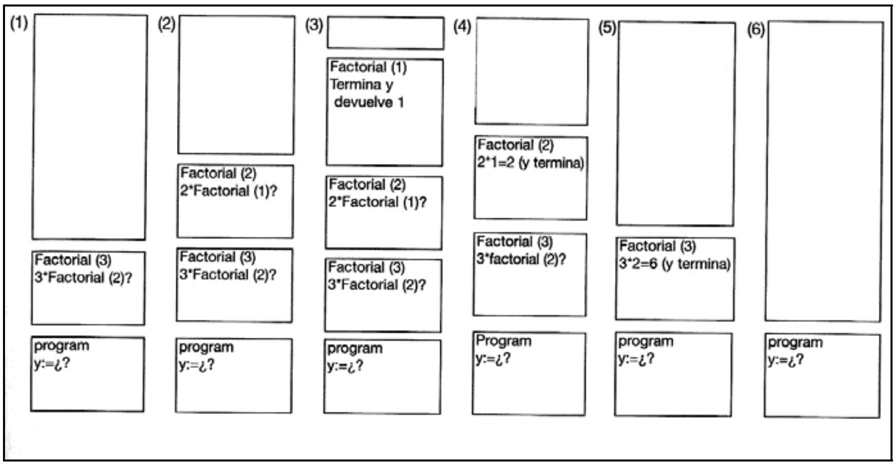


Ilustración 5, secuencia de pila durante el cálculo de factorial

4.5 Cálculo de la Serie de Números de Fibonacci

Para el cálculo de los números de la serie de Fibonacci, algoritmo presentado únicamente en [3], sólo se ha hallado un tipo de exposición:

- Explicación textual.

Sólo el texto y la fórmula matemática que lo resuelve han sido encontrados en la bibliografía para la ilustración de este problema de simplificación. Para este problema, a pesar de todo, resultaría considerablemente útil la representación del árbol de activación, que ayuda a ver cómo se van reduciendo los valores de entrada hasta alcanzar los valores base y apreciar cómodamente la dependencia de las llamadas, así como la redundancia de los cálculos que se realizan.

4.6 Otros Problemas Matemáticos

Existen otros problemas resolubles bajo la técnica "divide y vencerás" tales como la convolución de funciones o la transformada rápida de Fourier, disponibles en [10][11], para los que el tipo de exposición encontrada es únicamente:

- Explicación textual.

Esta explicación textual se encuentra acompañada de formulaciones y expresiones matemáticas que directamente ayudan a encontrar la solución del problema planteado pero no aportan ningún tipo de ilustración gráfica, ya sea esquemática o aplicada a un ejemplo concreto.

5. Algoritmos que Manejan Estructuras y Devuelven un Valor

Estos algoritmos manejan en todos los casos una estructura simple de datos, como un vector o matriz y retornan un valor como resultado de su ejecución que puede ser de cualquier tipo, pudiendo ser a su vez otro vector o matriz. En este grupo se encuentran los problemas de:

1) Búsqueda binaria.
2) Cálculo del máximo de un vector.
3) Cálculo del mínimo y máximo de un vector.
4) Selección (búsqueda del k-ésimo valor más pequeño).
5) Mediana del vector resultante de mezclar dos vectores ordenados
6) Búsqueda del elemento mayoritario de un vector.

El problema de la búsqueda binaria se presta, teniendo en cuenta la bibliografía revisada, a tres representaciones principalmente: el árbol binario balanceado con las posiciones del vector [5], una representación que muestra los sucesivos pasos y los valores que tienen los índices que delimitan la porción del vector que maneja el algoritmo en cada llamada recursiva [4] y una representación esquemática del vector que sutilmente indica la manera de proceder del algoritmo [14]. La segunda es muy útil para comprender cómo el algoritmo va desechando una mitad del vector restante basándose en los valores de la porción que maneja y en la premisa de que los valores están ordenados.

Las representaciones de los problemas de la búsqueda del valor máximo y de la búsqueda simultánea del valor máximo y mínimo de un vector son muy parecidas por motivos obvios. Una de ellas contiene el árbol de activación que representa las sucesivas divisiones del vector en un orden descendente y la transmisión de los valores resultantes en orden ascendente [11][13]. La representación se vale de flechas con el fin de facilitar la comprensión del sentido en que se desplaza cada elemento de la representación.

El problema de selección retorna el valor k-ésimo más pequeño de un vector no ordenado basándose en los mismos cálculos de pivote que utiliza el algoritmo Quicksort para la división del vector. Este algoritmo es ilustrado en algunas obras bibliográficas mediante representaciones secuenciales que muestran qué partes del vector se van manejando y cuáles se van desechando [4]. Además, en las representaciones del vector quedan distanciadas las posiciones del vector que contienen la posición del pivote y además aquellas que dejan de ser de interés para la búsqueda quedan ocultadas o atenuadas.

Por su parte, el cálculo de la mediana del vector resultante de mezclar dos vectores ordenados fue encontrado en una única obra bibliográfica, por lo que se cuenta con una única representación gráfica. Sorprende que la riqueza de este problema no sea explotada por más autores.

Por último, el algoritmo que comprueba la existencia de un valor mayoritario en un vector no cuenta con ninguna representación gráfica en la bibliografía manejada.

5.1 Búsqueda Binaria

Para la ilustración de la búsqueda binaria se han encontrado los siguientes tipos de representaciones:

- Explicación textual.
- Secuencia de los índices que limitan la parte tratada del vector.
- Árbol.
- Representación esquemática del vector

Este problema es uno de los más sencillos y de los que primero suele aparecer en las obras y en las clases de asignaturas de algoritmia en el tema de la técnica “divide y vencerás”. Aparece en varias obras de las consultadas [2][3][4][5][14][15] y con diferentes representaciones que son mostradas a continuación: la de la Ilustración 6 perteneciente a [4] y la de la Ilustración 7 a [14].

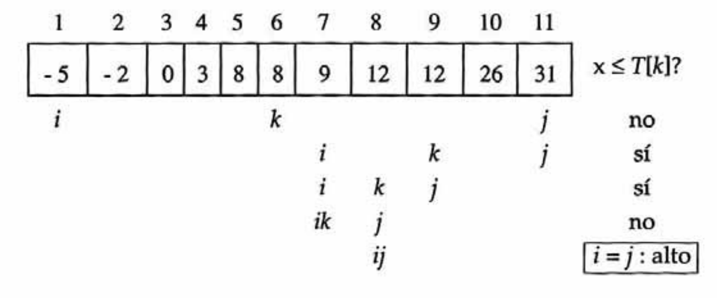


Figura 7.1. Búsqueda binaria de x = 12 en T[1..11]

Ilustración 6, secuencia de índices en la búsqueda binaria

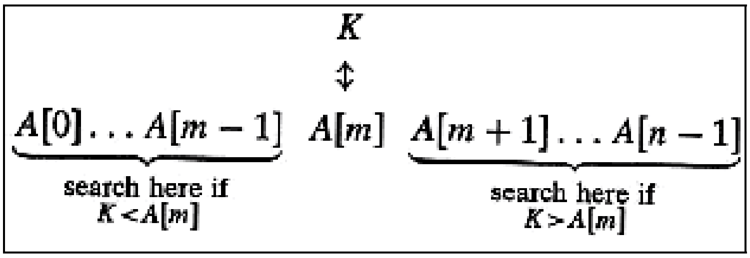


Ilustración 7, esquema sobre cómo divide el vector la búsqueda binaria

5.2 Máximo de un Vector

Para la ilustración de la búsqueda del máximo de un vector se ha recopilado el siguiente tipo de representación:

- Árbol completo secuencial.

Este problema es también uno de los más fáciles de aprender dentro de la técnica "divide y vencerás". Tan sólo un autor [11] lo insertó en su obra bibliográfica, aportando la representación gráfica que queda recogida mediante la Ilustración 8, donde se puede apreciar el árbol expresa cómo se divide el vector y mediante flechas cómo se propagan los resultados parciales hasta obtener el resultado final.

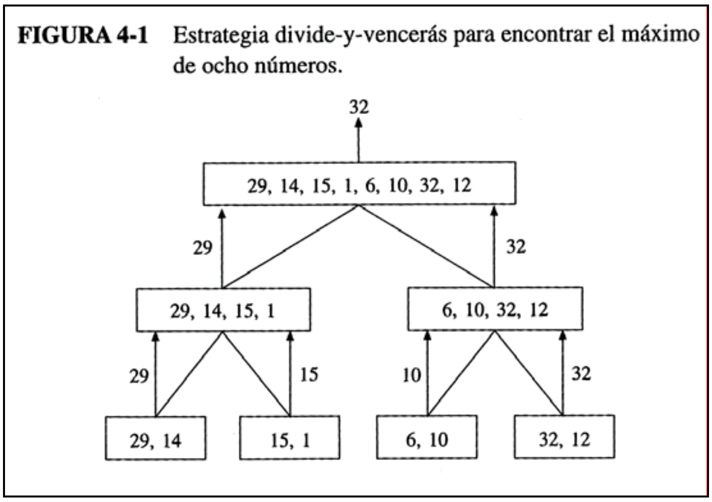


Ilustración 8, árbol direccionado para hallar el máximo de un vector

5.3 Mínimo y Máximo de un Vector

Para la ilustración de la búsqueda del máximo de un vector se han encontrado dos maneras de presentarlo:

- Árbol con símbolos.
- Representación textual

Este problema es de mayor complejidad que los dos anteriores, ya que necesita usar una estructura de datos donde almacenar los dos valores que busca al mismo tiempo. Únicamente dos autores [13][15] lo insertaron en su obra bibliográfica. El primero de ellos aportó dos representaciones que toman como base un árbol de llamadas. Tales dos representaciones quedan mostradas a continuación mediante la Ilustración 9 y la Ilustración 10.

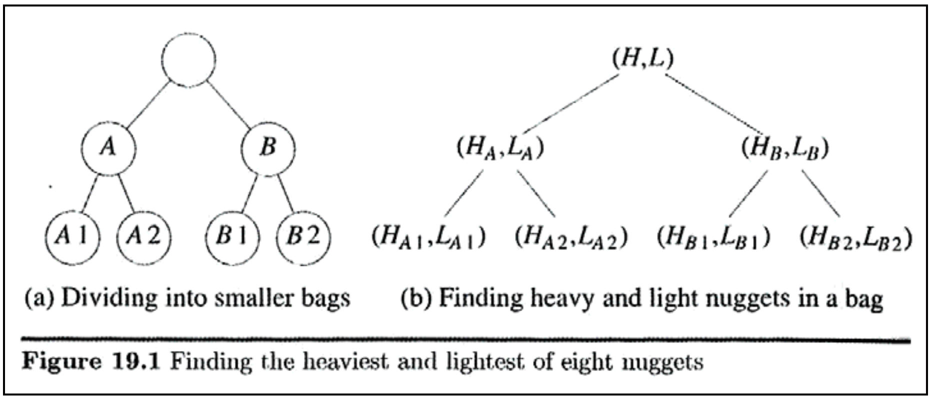


Ilustración 9, esquema sobre el cálculo del máximo y mínimo de un vector

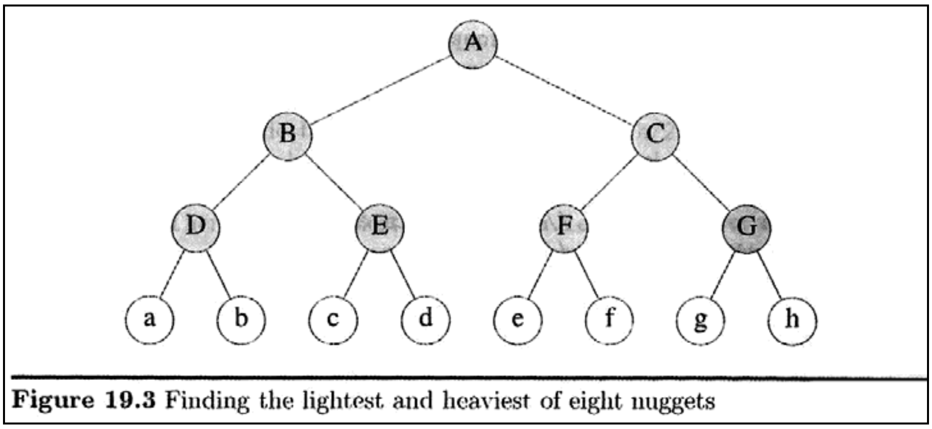


Ilustración 10, esquema sobre el cálculo del máximo y mínimo de un vector

5.4 Selección (k-ésimo Valor más Pequeño)

El algoritmo de selección cuenta con dos tipos de representaciones en la bibliografía consultada:

- Representación textual.
- Secuencia del vector, con realzado de partes importantes.

Cinco autores [1][2][3][4][5] insertaron este problema en sus obras, pero sólo en cuatro de esas obras [1][2][4][5] aportaron alguna representación gráfica, que son las que se representan a continuación, respectivamente, en la Ilustración 14, que representa una agrupación de los elementos del vector adecuadamente etiquetados, la Ilustración 12, que da una explicación general del problema en una representación espacial de dos dimensiones, la Ilustración 11, que muestra una secuencia de partición del vector, y la Ilustración 13, que representa la misma agrupación de elementos que la Ilustración 14 pero empleando otros elementos gráficos.

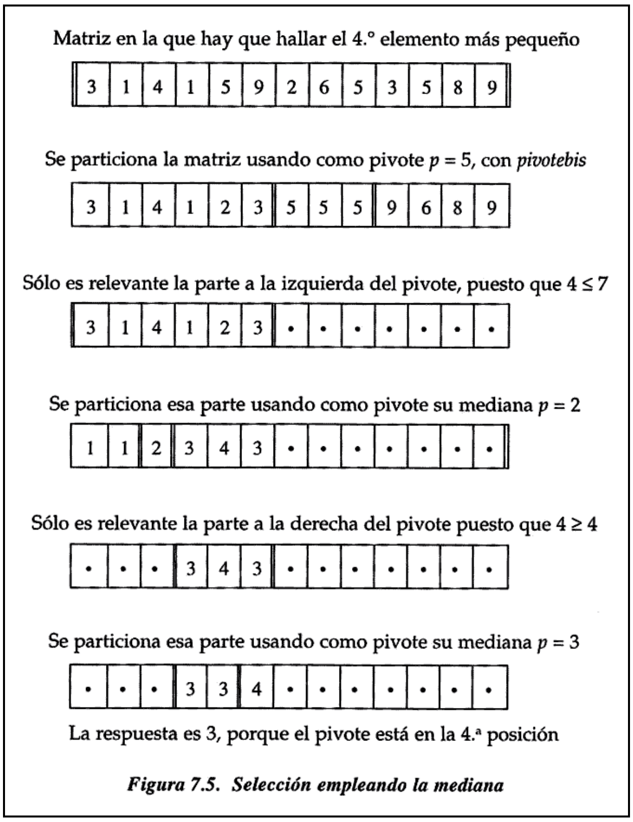


Ilustración 11, secuencia del algoritmo de selección

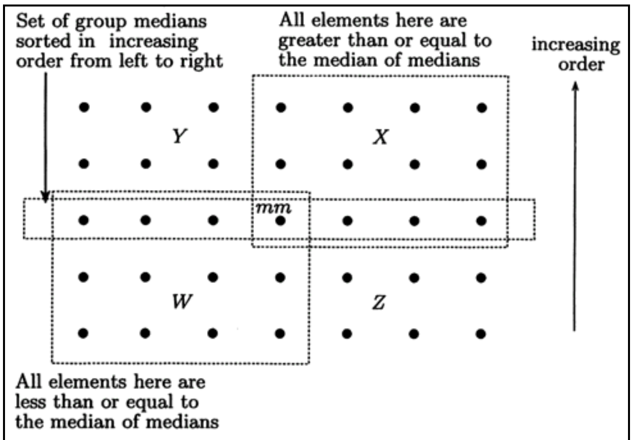


Fig. 6.2 Analysis of Algorithm SELECT.

Ilustración 12, esquema gráfico del algoritmo de selección en 2D

Chapter 9 Medians and Order Statistics

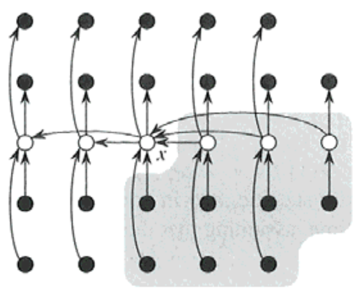


Figure 9.1 Analysis of the algorithm SELECT. The n elements are represented by small circles, and each group occupies a column. The medians of the groups are whitened, and the median-of-medians x is labeled. (When finding the median of an even number of elements, we use the lower median.) Arrows are drawn from larger elements to smaller, from which it can be seen that 3 out of every full group of 5 elements to the right of x are greater than x, and 3 out of every group of 5 elements to the left of x are less than x. The elements greater than x are shown on a shaded background.

Ilustración 13, esquema gráfico del algoritmo de selección por agrupación

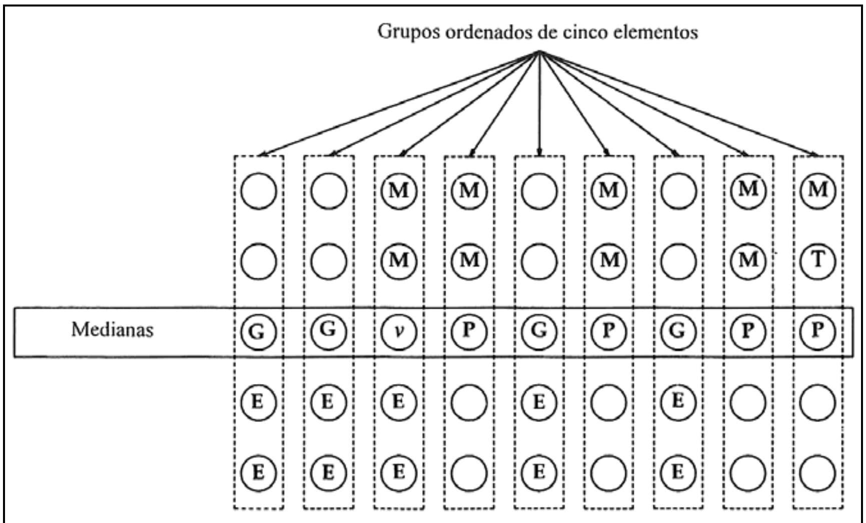


Figura 10.36 Cómo se elige el pivote

Ilustración 14, esquema gráfico del algoritmo de selección por agrupación

5.5 Mediana del Vector Resultante de Mezclar Dos Vectores Ordenados

El problema para hallar la mediana del vector resultante de mezclar dos vectores ordenados fue encontrado en una única obra bibliográfica, [12], que ofrece la siguiente representación:

- Representación estructural de los vectores.

Este algoritmo no es utilizado apenas por los autores, si bien se considera que es de un valor añadido frente a otros al manejar simultáneamente dos estructuras de datos (vectores) aplicando sobre ellas la técnica "Divide y vencerás" para lograr hallar un valor significativo.

Se presenta a continuación en la Ilustración 15 la manera de [12] de presentar el algoritmo mediante una explicación genérica.

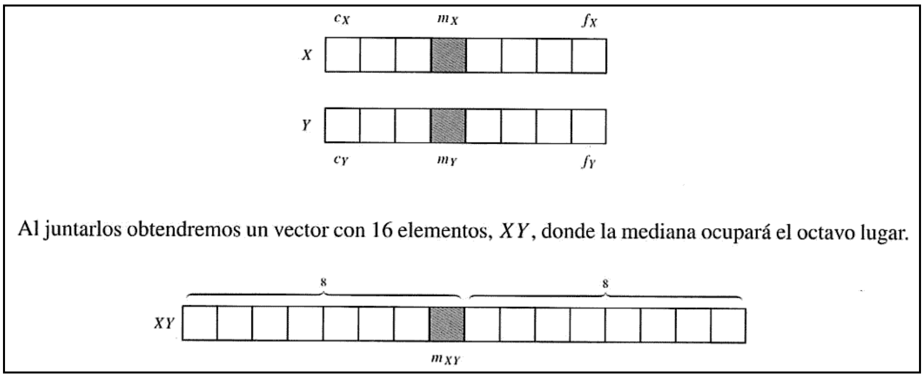


Ilustración 15, explicación para hallar la mediana un vector mezcla

5.6 Elemento Mayoritario de un Vector

El problema de la búsqueda del elemento mayoritario de un vector sólo cuenta con un tipo de representación:

- Representación textual.

En la bibliografía consultada sólo aparece explicado mediante la tradicional exposición textual [3]. No obstante, el algoritmo se podría prestar a ser ilustrado mediante una representación secuencial parecida a la de algoritmos como Selección o Quicksort, permitiendo que se visualizara cómo se van desechando parte de los dos vectores de entrada hasta encontrar la posición que quedaría situada en la posición de la mediana en el vector resultante de mezclar ambos.

6. Algoritmos que Manejan y Modifican Estructuras

Esta categoría contiene varios algoritmos que manejan una estructura simple de datos, como un vector o una matriz, pero sin aportar tras su finalización un valor de retorno, pues la ejecución modifica el contenido de la citada estructura, cuyo estado final puede ser recogido como valor de retorno de propio algoritmo. Quedan recogidos aquí algoritmos como:

1) Ordenación de vectores Mergesort.
2) Ordenación de vectores Quicksort.
3) Coloreado de tablero defectuoso.
4) Intercambio de partes en un vector.
5) Ordenación de alumnos según altura para que no queden tapados al ver dos pizarras en dos paredes adyacentes

El algoritmo Mergesort centraliza la mayor parte de sus operaciones en la tarea de combinación de resultados. Se ha de recordar que mezcla los resultados parciales obtenidos previamente en las subllamadas recursivas que se realizan sobre cada mitad del vector. Este proceso de combinación no suele ilustrarse habitualmente en las obras bibliográficas por ser muy fácil de comprender conceptualmente. Para el algoritmo habitualmente se emplea el árbol de activación, mostrando el estado inicial y el estado final en cada subllamada [2] o bien un grafo que se ramifica hasta problemas elementales que van aumentando de tamaño a medida que van quedando resueltos hasta quedar resuelto el vector [14] o, por último, una secuencia de representaciones del vector que van mostrando paso a paso cómo va modificándose el interior del vector según lo va ordenando el algoritmo [4][9][13].

Por otro lado, el algoritmo Quicksort centra sus esfuerzos en las tareas de división del vector, calculando la posición del pivote y dejándolo situado en su posición final. Al ser éste un procedimiento de cierta complejidad conceptual, suele ser representado en la bibliografía paso a paso para lograr un completo entendimiento. Estas representaciones secuenciales reflejan cómo se van comparando e intercambiando valores y cómo se el pivote queda ubicado en la que será su posición final tras la ejecución del algoritmo haciendo uso de flechas para representar a los índices y de distintas convenciones (coloreado, doble separación) para remarcar unas partes del vector frente a otras [4][5][14].

Como sucede con el algoritmo Quicksort, el problema de colorear un tablero defectuoso con figuras en forma de L registra en la tarea de división la mayor parte de su trabajo. Será en ese momento del algoritmo cuando se localice el defecto y se coloreen adecuadamente los demás cuadrantes de la matriz para poder dividirla y obtener subproblemas equivalentes al problema originalmente planteado, pero de menor tamaño. Los autores coinciden en representar la matriz con un elemento en forma de L [13] o con varios [9] para explicar cómo es el proceso de rellenado.

Por su parte, el algoritmo de desplazamiento de las posiciones de un vector es bastante sencillo de comprender y fácil de representar, diferenciando mediante alguna separación las partes que se van a desplazar de las que permanecen fijas [3]. Además, se suelen utilizar flechas para representar los índices que se manejan con el fin de recorrer el vector para ir moviendo los elementos.

Para finalizar, se presenta el algoritmo que tiene como objetivo ordenar los alumnos de un aula en función de su altura para que no queden estorbados a la hora de mirar a la pizarra. El algoritmo tiene una segunda fase en la que se añade una segunda pizarra en una pared lateral y éste debe aprovechar el ordenamiento previo para reordenar los alumnos de forma que no se estorben al observar ninguna de las dos pizarras. Este algoritmo es presentado en [8].

6.1 Mergesort

El algoritmo Mergesort (ordenación por mezcla) es uno de los más conocidos algoritmos de ordenación. Las representaciones encontradas para ilustrarlo son las siguientes:

- Representación textual.
- Árbol de activación, con flechas y numeración secuencial.
- Secuencia de la tarea de combinación.
- Secuencia de la partición y unión del vector.
- Secuencia de la división del vector.

No extraña el alto número de representaciones encontradas, dada su popularidad y su consecuentemente alto número de obras en las que aparece [1][2][3][4][5][9][10][13][14].

La primera de las que se ofrece a continuación en la Ilustración 16, extraída de [4], representa la definición inductiva del algoritmo, ilustrando la separación del mismo en dos mitades que, tras su ordenación parcial, se funden en un nuevo vector ordenado. La segunda de ellas (Ilustración 17), de [13], representa el proceso de mezcla, con las divisiones del vector ya realizadas. Poco a poco, lo que inicialmente son posiciones aisladas van dando forma a conjuntos de vectores ordenados que finalmente, gracias al citado proceso de combinación, se mezclan en uno solo.

La tercera representación listada gracias a la Ilustración 18, perteneciente a [9], muestra un ejemplo entero, en la misma línea de la Ilustración 16, pero sin quedarse en el segundo nivel de recursión. Esta representación se puede representar unida con ramas, como en [14], recogida en la Ilustración 19. La quinta representación, de [10], es una representación genérica de la división de los vectores en mitades que se centra en el cálculo de la eficiencia del algoritmo, queda representada en la Ilustración 20. Por su parte, la sexta (Ilustración 21), extraída de [2], muestra en un formato similar al árbol de recursión qué entradas toma cada subllamada recursiva y qué subsector ordenado devuelve. Las flechas ayudan a entender el sentido en el que se trasladan los datos. Por su parte, la Ilustración 22 tomada desde [5] se centra en la parte de recombinación de soluciones mediante un árbol en el que se refleja cómo los resultados simples se van recombinando mediante operaciones de mezcla, indicadas en el árbol. Para finalizar el repaso por las representaciones del algoritmo, se presenta la Ilustración 23 de [1] que explica mediante un ejemplo el proceso de mezcla, no encontrado en ninguna otra obra bibliográfica de las consultadas y la Ilustración 24, de [12], que muestra árboles preparados para la ordenación por mezcla.

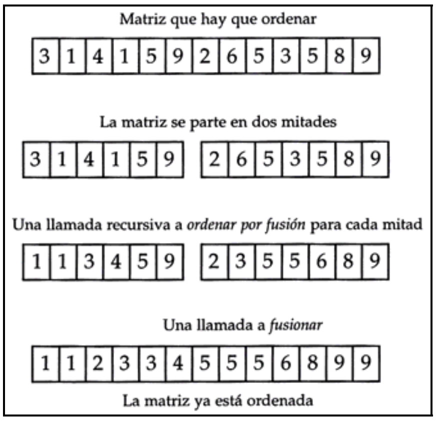


Ilustración 16, definición recursiva de Mergesort

initial segments	[8]	[4]	[5]	[6]	[2]	[1]	[7]	[3]
merge to b	[4	8]	[5	6]	[1	2]	[3	7]
copy to a	[4	8]	[5	6]	[1	2]	[3	7]
merge to b	[4	5	6	8]	[1	2	3	7]
copy to a	[4	5	6	8]	[1	2	3	7]
merge to b	[1	2	3	4	5	6	7	8]
copy to a	[1	2	3	4	5	6	7	8]

Figure 19.9 Merge sort example

Ilustración 17, secuencia de Mergesort (flujo pasivo)

Example 5.2.4. We show how mergesort sorts the array

| 12 | 30 | 21 | 8 | 6 | 9 | 1 | 7 | .

The array is first divided into two equal parts

| 12 | 30 | 21 | 8 | | 6 | 9 | 1 | 7 | .

Each part is then sorted by mergesort. The process begins by dividing each part into equal parts

| 12 | 30 | | 21 | 8 | | 6 | 9 | | 1 | 7 |

and then each of these parts into equal parts

| 12 | | 30 | | 21 | | 8 | | 6 | | 9 | | 1 | | 7 | .

This subdividing process now ends because each part contains only one item. Each pair is then merged

| 12 | 30 | | 8 | 21 | | 6 | 9 | | 1 | 7 | .

Each of these pairs is then merged

| 8 | 12 | 21 | 30 | | 1 | 6 | 7 | 9 | .

Finally these pairs are merged

| 1 | 6 | 7 | 8 | 9 | 12 | 21 | 30 |

to obtain the sorted array.

Ilustración 18, secuencia de Mergesort (flujo activo y pasivo)

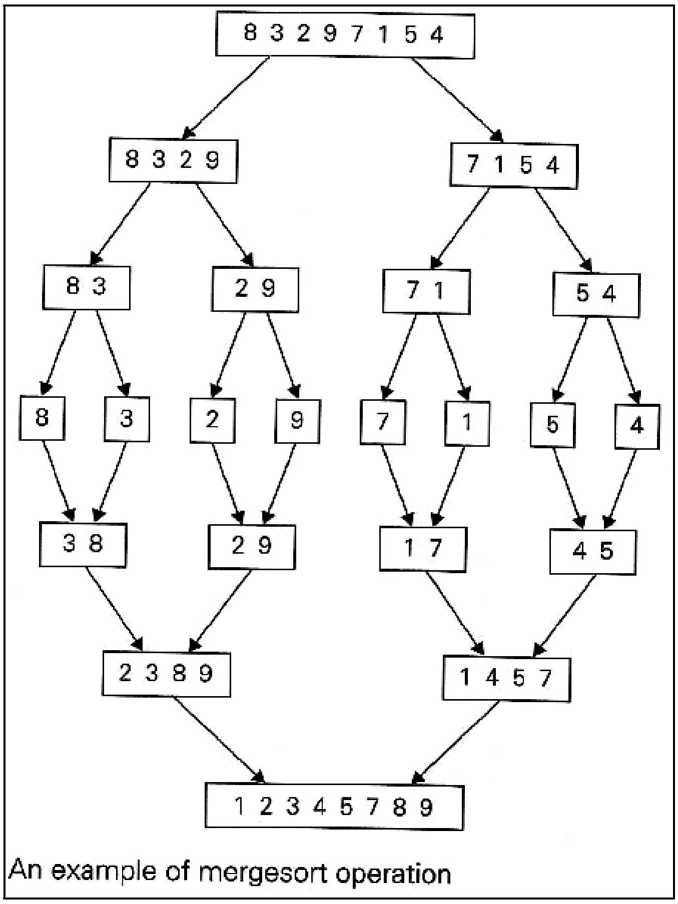


Ilustración 19, secuencia de Mergesort (flujo activo y pasivo)

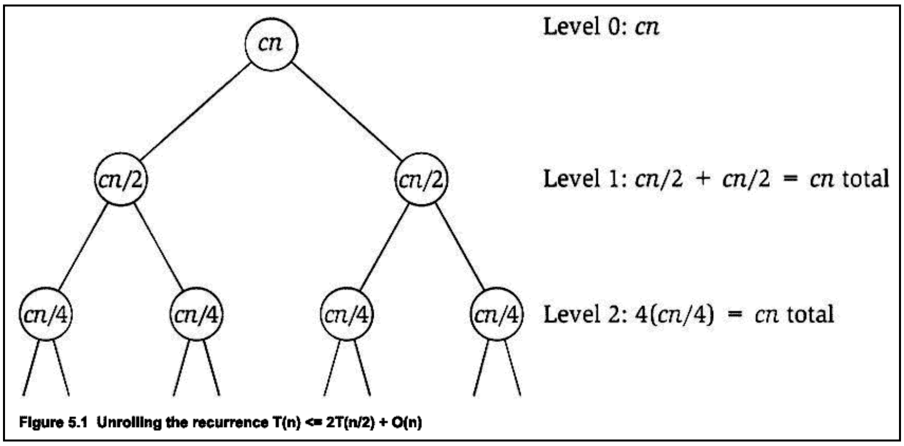


Ilustración 20, árbol esquemático para calcular la complejidad

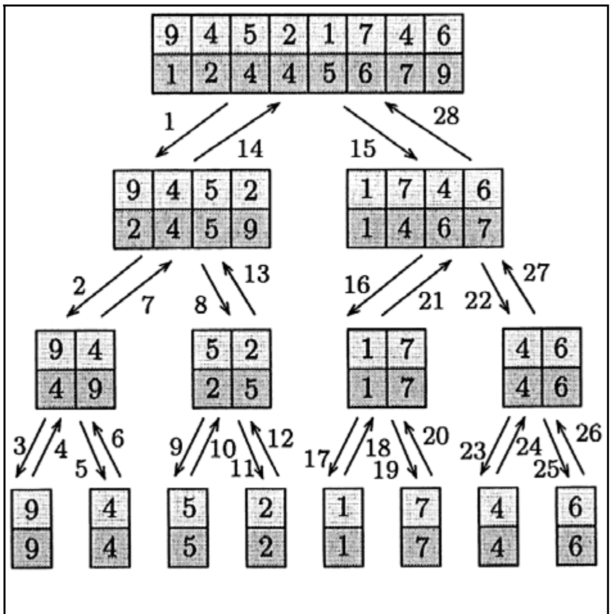


Fig. 6.1 The behavior of Algorithm MERGESORT.

Ilustración 21, árbol de activación de Mergesort

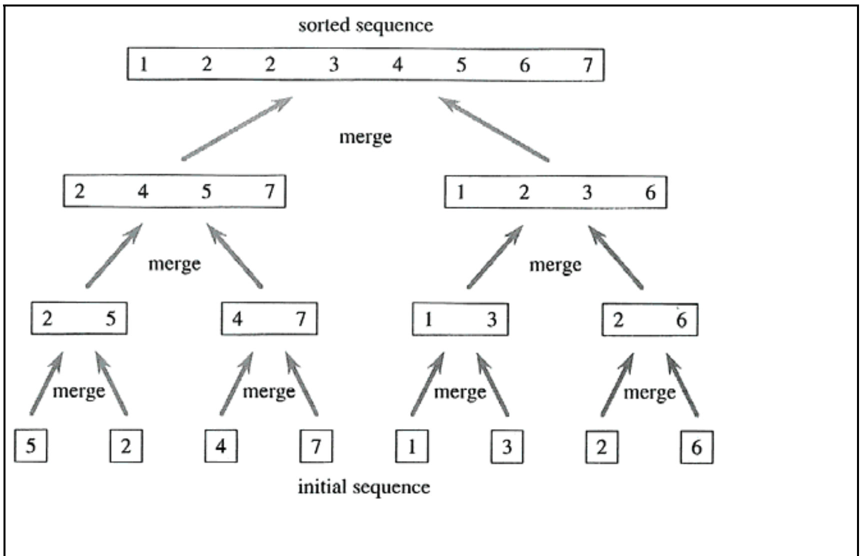
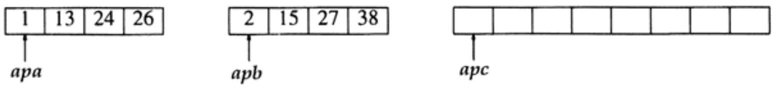
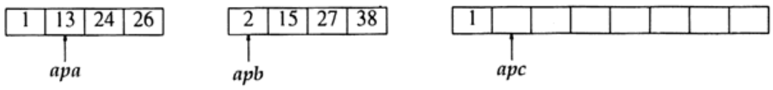


Figure 2.4 The operation of merge sort on the array $A = \langle 5, 2, 4, 7, 1, 3, 2, 6 \rangle$. The lengths of the sorted sequences being merged increase as the algorithm progresses from bottom to top.

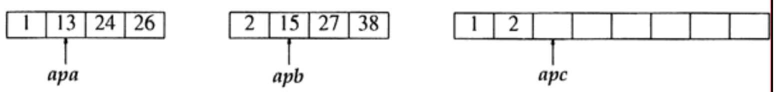
Ilustración 22, árbol de recursión de Mergesort con mezclas indicadas



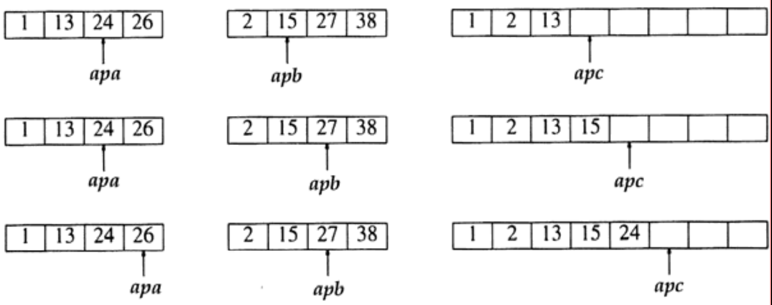
Si el arreglo *a* contiene 1, 13, 24, 26 y *b* contiene 2, 15, 27, 38, el algoritmo procede como sigue: primero, se hace una comparación entre 1 y 2. 1 se agrega a *c*, y después se comparan 13 y 2.



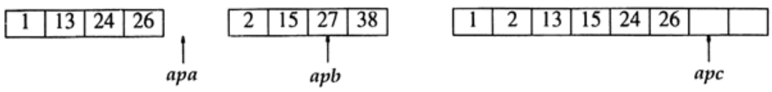
2 se agrega a *c*, y luego se comparan 13 y 15.



13 se agrega a *c*, y después se comparan 24 y 15. Esto continúa hasta comparar 26 y 27.



26 se agrega a *c*, y se agota el arreglo *a*.



El resto del arreglo *b* se copia en *c*.

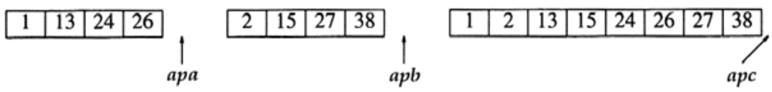


Ilustración 23, proceso de mezcla de Mergesort

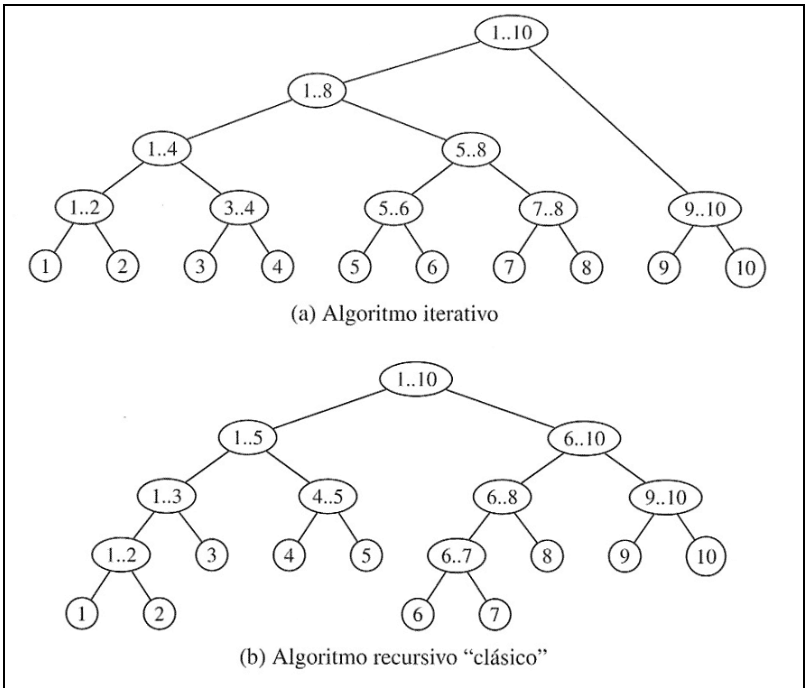


Figura 11.3: Árboles de etapas para la ordenación por mezclas.

Ilustración 24, árboles para la ordenación por mezcla

6.2 Quicksort

Al igual que mergesort, quicksort es un conocido algoritmo de ordenación que no falta en ningún curso de algoritmia e introducción a la programación, por lo que su difusión a través de la bibliografía es bastante recurrente y amplio [1][2][3][4][5][13][14]. Las representaciones encontradas para él son tres:

- Representación textual.
- Secuencia del vector marcando las divisiones y la posición del pivote.
- Esquema con el vector dividido.

Gran parte de las representaciones encontradas se centran en la tarea de división del vector, que deja ubicado en su posición final al valor que actúa como pivote.

A continuación se muestra la Ilustración 25 extraída de [4], donde se puede ver de manera secuencial cómo se calcula la posición del pivote para la posterior división del vector. El mismo proceso se visualiza mediante un ejemplo en la Ilustración 26, de [5]. A continuación, el mismo autor expone una explicación genérica del algoritmo de manera gráfica, recogido en la Ilustración 27.

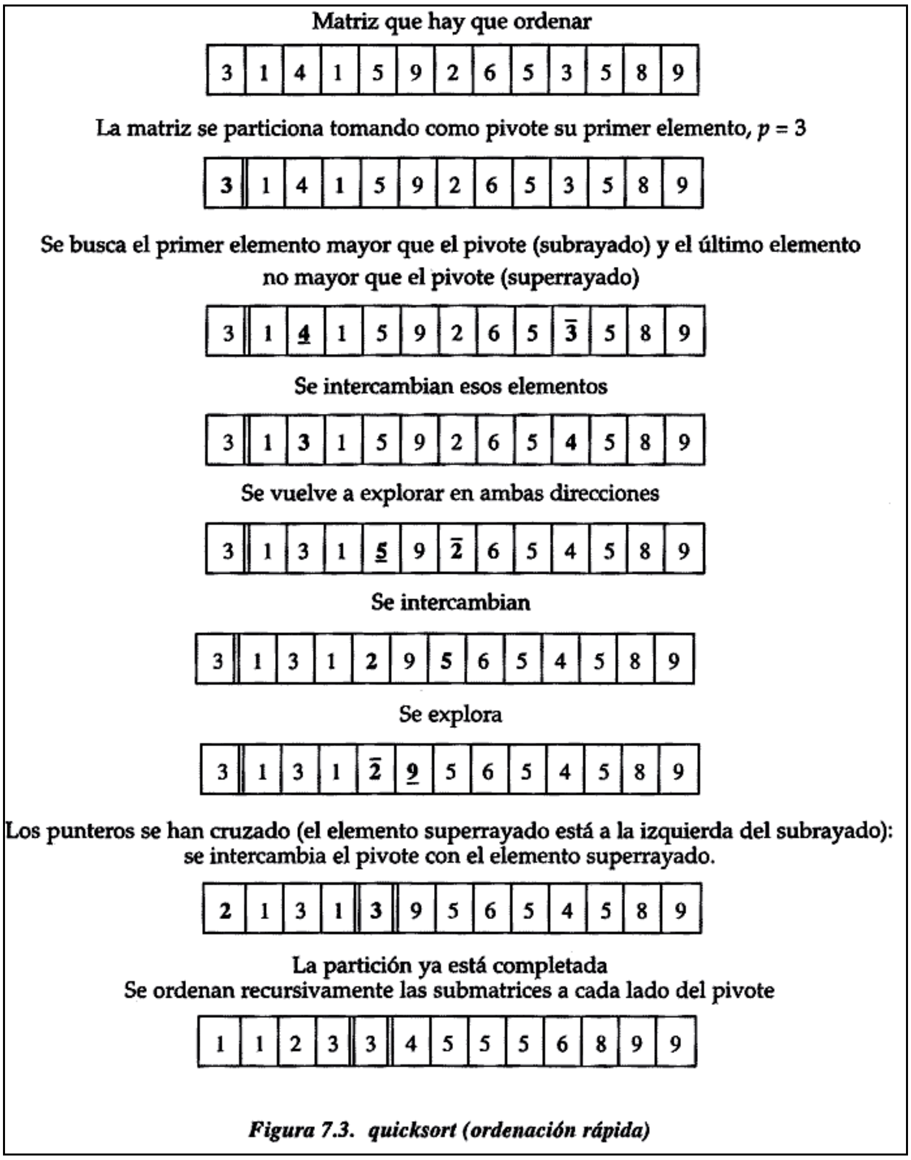


Ilustración 25, secuencia de división del vector en Quicksort

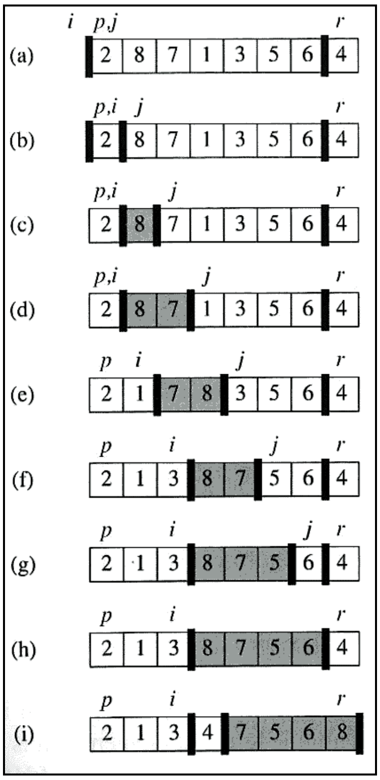


Ilustración 26, secuencia de división del vector en Quicksort

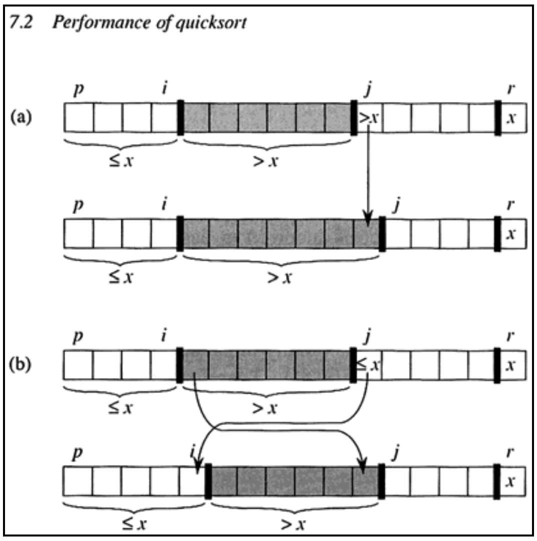


Ilustración 27, esquema del funcionamiento de Quicksort

A continuación se exponen en la Ilustración 28 y en la Ilustración 29 las representaciones esquemáticas de la división del vector propuestas por [2][14], respectivamente. Posteriormente se muestra la Ilustración 30, de [14], que refleja el proceso secuencial de reubicación de los datos para poder llevar a cabo la partición del vector. Tras ella aparece la Ilustración 31, del mismo autor, que deja ver el árbol de llamadas donde se deja constancia de los parámetros de partición y de los pivotes que permiten partir el vector en sucesivas llamadas recursivas. Por último, se presenta la Ilustración 32, de [1], donde el autor expresa de una manera muy visual y explícita la no ordenación de los contenidos del vector gracias a la representación de sus distintas partes como conjuntos.

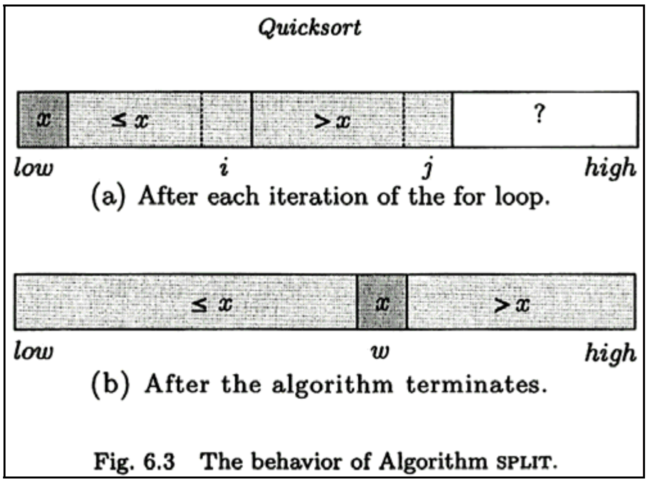


Ilustración 28, esquemas del funcionamiento de Quicksort

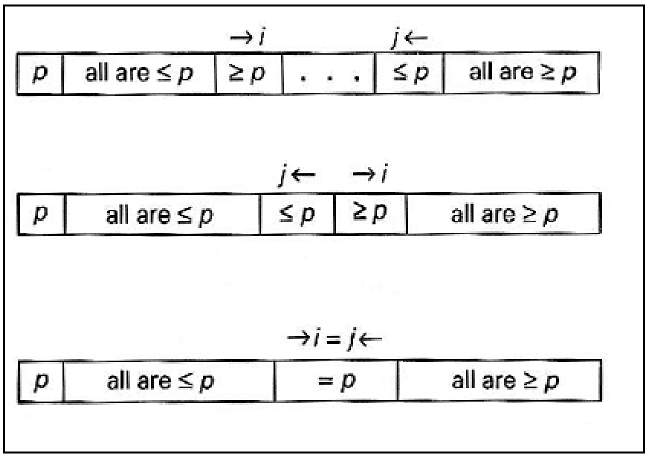


Ilustración 29, esquemas del funcionamiento de Quicksort

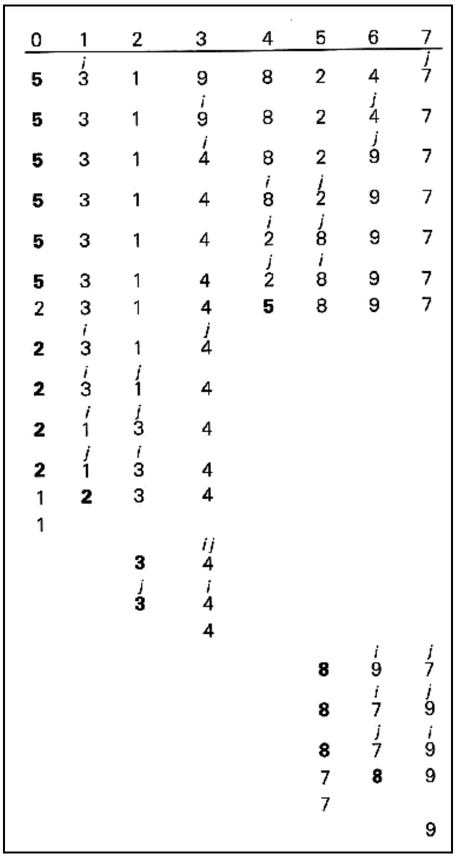


Ilustración 30, proceso cronológico de Quicksort con índices indicados

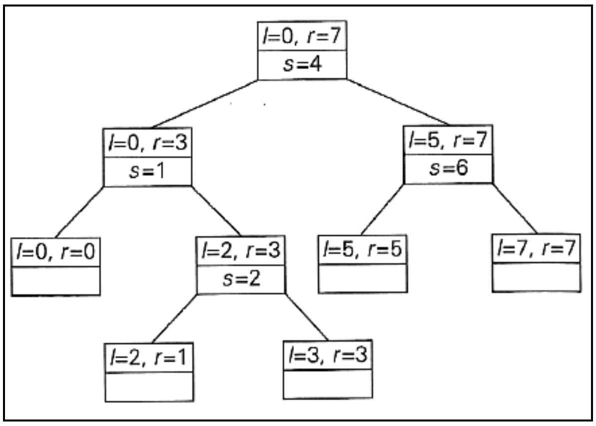


Ilustración 31, árbol que representa las llamadas recursivas de Quicksort

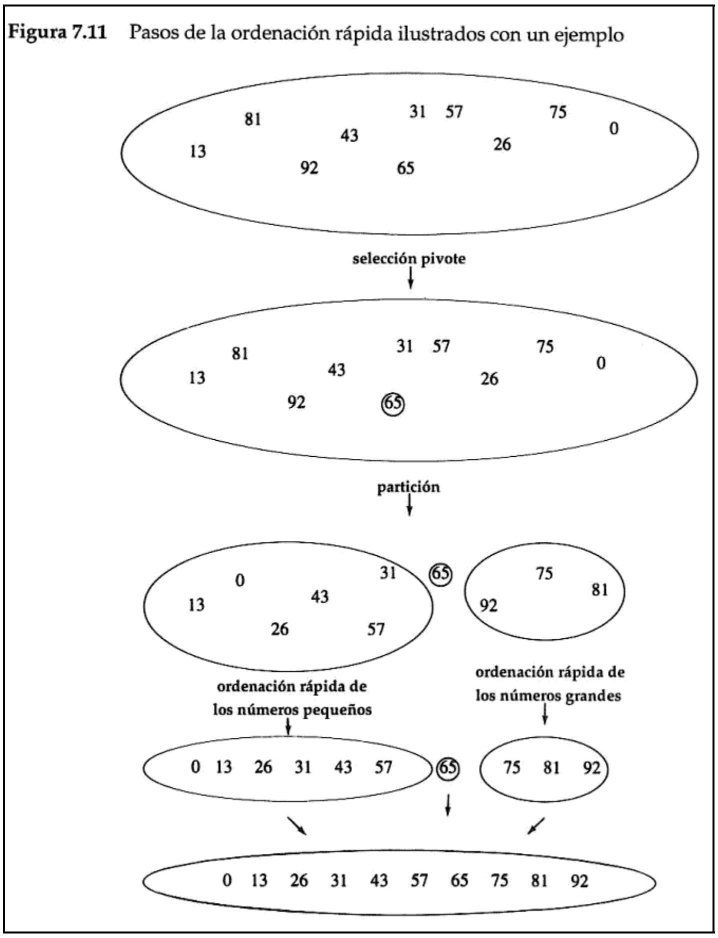


Ilustración 32, representaciones del vector sobre el que se aplica Quicksort

6.3 Coloreado de Tablero Defectuoso

Este algoritmo ha sido encontrado en dos obras [9][13] y ambas lo ilustran de manera similar, haciendo uso de representaciones gráficas:

- Tablero con uno o más triominós.
- Tablero sobre el que se dibujan las divisiones que se realizan.

Ambos libros dan un tratamiento similar mediante las representaciones que se recogen a continuación. La Ilustración 33, de [13], expresa de una manera certera la idea de que se deben colorear los cuadrantes que no tienen el cuadro defectuoso para poder efectuar la división de la matriz y seguir la ejecución del algoritmo. La dos siguientes (Ilustración 34 e Ilustración 35), extraídas desde [9], expresan de manera secuencial a distintos niveles cómo funciona el algoritmo.

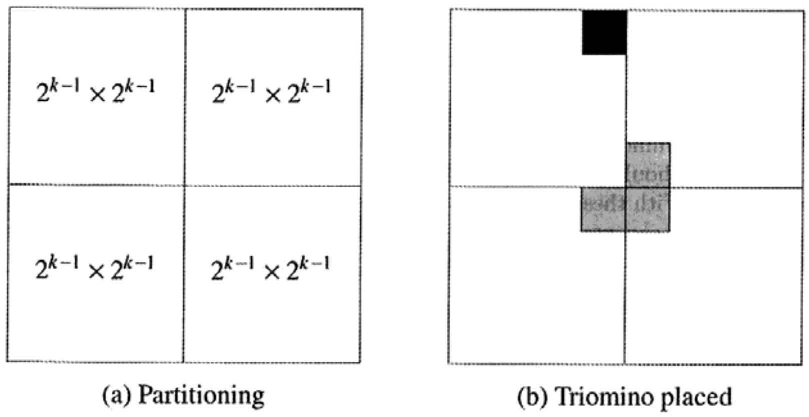


Figure 19.6 Partitioning a $2^k \times 2^k$ board

Ilustración 33, esquema y primera división del tablero defectuoso

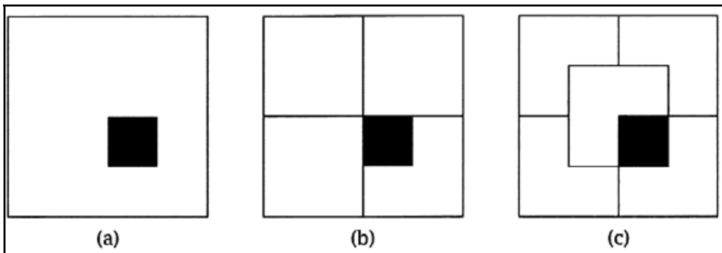


Figure 5.1.5 Tiling the deficient 4×4 board shown in (a). First, the board is divided into four 2×2 subboards as shown in (b). The subboard that contains the missing square is recursively tiled; in this case, the deficient 2×2 board is a tromino. Next, we place a tromino as shown in (c) so that each of its 1×1 squares lies in one of the three remaining subboards. Each of these 1×1 squares is considered as missing in the remaining subboards. We can then recursively tile these deficient subboards. Again, each of the deficient 2×2 boards is a tromino, so the problem is solved. □

Ilustración 34, primera y segunda división del tablero defectuoso

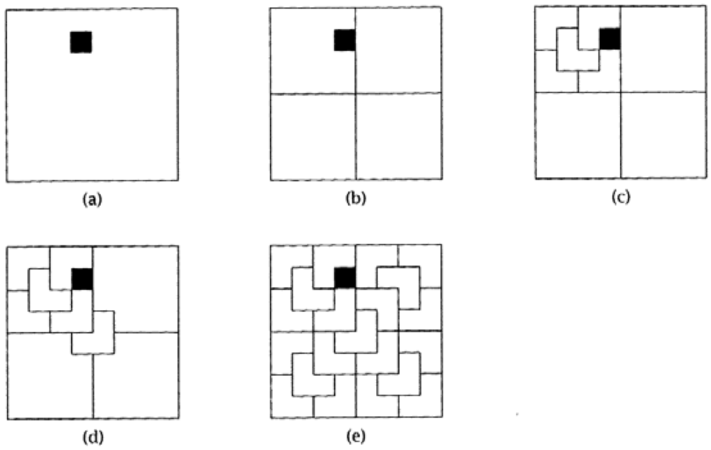


Figure 5.1.6 Tiling the deficient 8 × 8 board shown in (a). First, the board is divided into four 4 × 4 subboards as shown in (b). The subboard that contains the missing square is recursively tiled as shown in (c). Next, we place a tromino as shown in (d) so that each of its 1 × 1 squares lies in one of the three remaining subboards. Each of these 1 × 1 squares is considered as missing in the remaining subboards. We can then recursively tile each of these deficient 4 × 4 subboards as shown in (e). The problem is solved.

Ilustración 35, división del tablero defectuoso en distintos momentos

6.4 Intercambio de Partes en un Vector

El problema del intercambio de una parte en un vector, también resoluble bajo la técnica de “divide y vencerás”, fue encontrado en una única obra bibliográfica [3], con dos representaciones:

- Esquema con sombreado en las zonas de movimiento
- Secuencia con índices apuntando a los límites de las zonas manejables.

A continuación se muestran tales representaciones en el respectivo orden mediante la Ilustración 36 y la Ilustración 37.

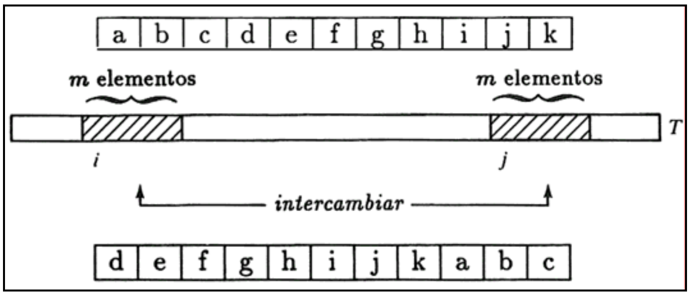


Ilustración 36, esquema del intercambio de partes de un vector

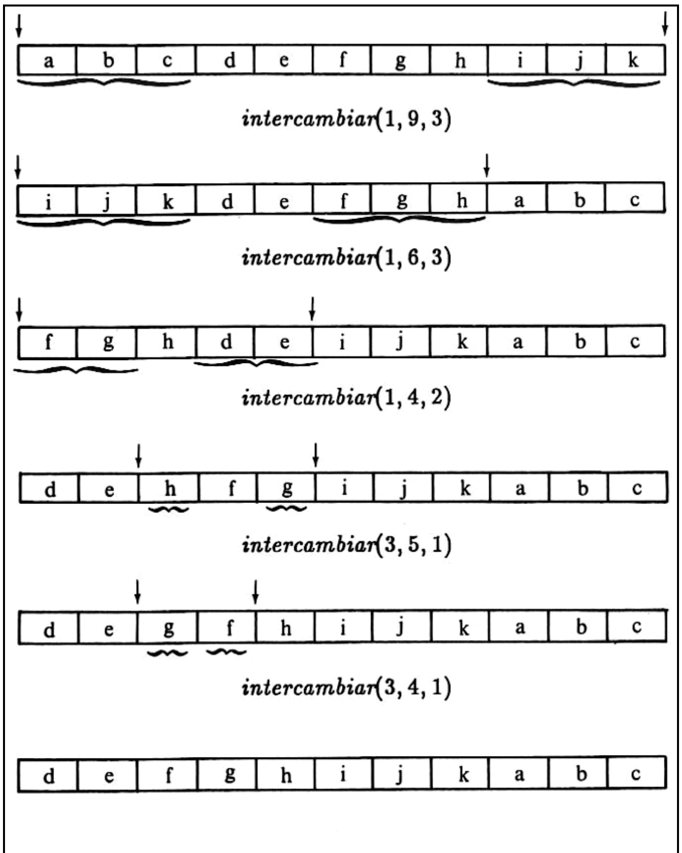


Figura 4.10.2

Ilustración 37, secuencia del intercambio de partes de un vector

7. Algoritmos Geométricos

Esta categoría engloba algoritmos que tienen un carácter marcadamente geométrico, y que hacen uso de planos, puntos, líneas u otros elementos geométricos. Algunos de los algoritmos que pertenecen a esta clase son:

1) Cálculo del par de puntos más cercanos entre sí dado un conjunto de puntos en el plano.
2) Cálculo del número de puntos que domina cada punto del plano dado.
3) Cálculo de polígonos convexos.
4) Cálculo de diagramas de Voronoi.

Todos estos algoritmos son representados habitualmente haciendo uso de ejemplos donde se sitúan puntos u otros elementos en un eje de coordenadas de dos dimensiones [1][2][5][9][10][11][13][14], lo que proporciona una imagen directa del problema. En estas representaciones se pueden emplear zonas sombreadas, coloreadas, remarcadas, o separadas por líneas o flechas que sirven para indicar cálculos, relaciones, divisiones del problema, etc.

7.1 Par de Puntos más Cercanos entre sí en un Conjunto

Este problema fue encontrado en dos de las obras consultadas [9][13] y en ambas se encontró una representación:

- Representación espacial (dependiente del dominio).

En cada libro la representación es diferente, empleando zonas sombreadas o delimitaciones mediante líneas, pero la esencia de ambas es idéntica. Se muestran a continuación dos imágenes de [13] (Ilustración 38 e Ilustración 39), otras dos posteriormente de [9] (Ilustración 40 e Ilustración 41) y una de [14] (Ilustración 42), junto con una última, la Ilustración 46, de [11]. Conviene resaltar las similitudes en la utilización de los elementos (líneas reparatorias, ausencia de rellenos) dentro de la Ilustración 41, Ilustración 42 y la Ilustración 46.

En las dos primeras se puede apreciar cómo el autor hace uso del sombreado para realzar la parte del espacio de interés para el algoritmo mientras que en las dos siguientes se emplean líneas para la separación de zonas con diferentes intereses. Las primera y tercera imágenes muestran el estado inicial del problema.

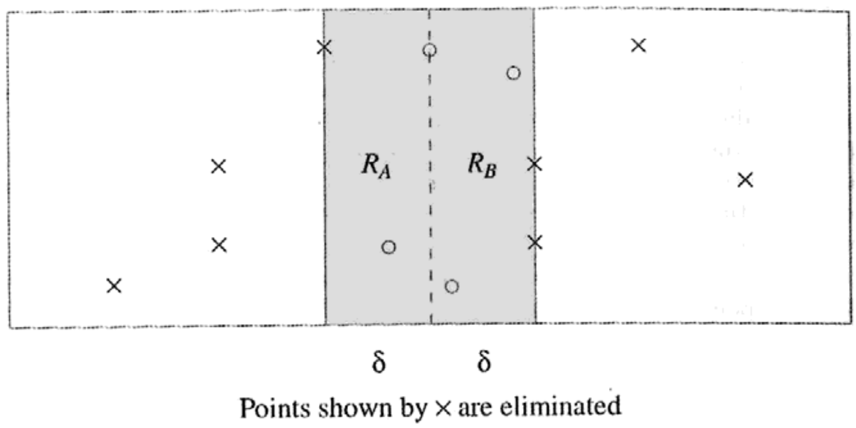


Figure 19.16 Eliminating points that are too far from the dividing line

Ilustración 38, representación espacial completa (par de puntos)

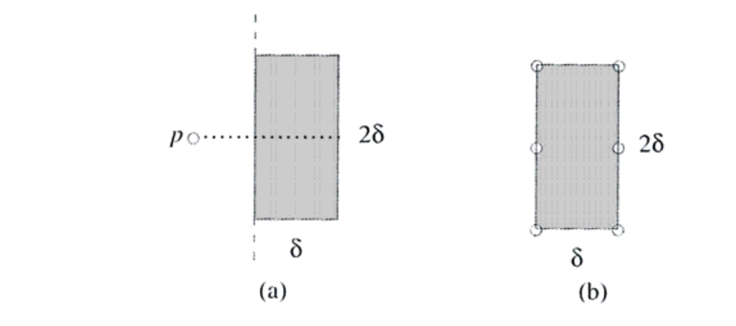


Figure 19.17 Comparing region for p

Ilustración 39, representación espacial restringida (par de puntos)

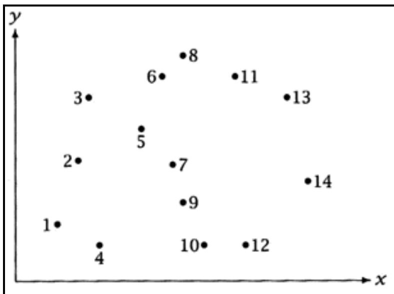


Figure 5.3.1 Types of plywood. Each type of plywood is plotted as a point in the plane; the x-coordinate measures the number of knotholes per square foot, and the y-coordinate measures the smoothness. The closest pair of points (6 and 8) represents the most similar types of plywood.

Ilustración 40, representación espacial con ejes (par de puntos)

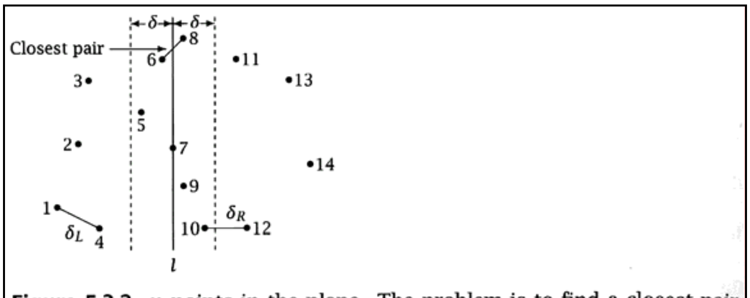


Figure 5.3.2 n points in the plane. The problem is to find a closest pair. For this set, the closest pair is 6 and 8. Line l divides the points into two approximately equal parts. The closest pair in the left half is 1 and 4, which is δ_L apart. The closest pair in the right half is 10 and 12, which is δ_R apart. Any pair (e.g., 6 and 8) closer together than $\delta = \min\{\delta_L, \delta_R\}$ must lie in the vertical strip of width 2δ centered at l.

Ilustración 41, representación espacial completa (par de puntos)

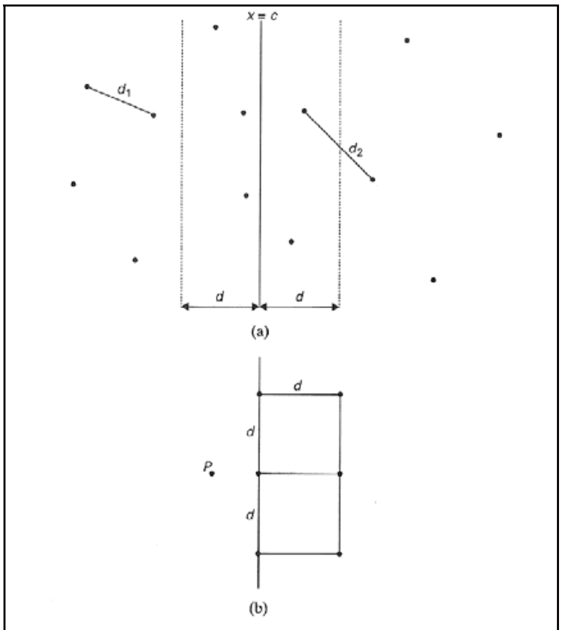


FIGURE 4.6 (a) Idea of the divide-and-conquer algorithm for the closest-pair problem. (b) The six points that may need to be examined for point P.

Ilustración 42, representaciones espaciales completa y selectiva (par de puntos)

FIGURA 4-4 El problema del par más cercano.

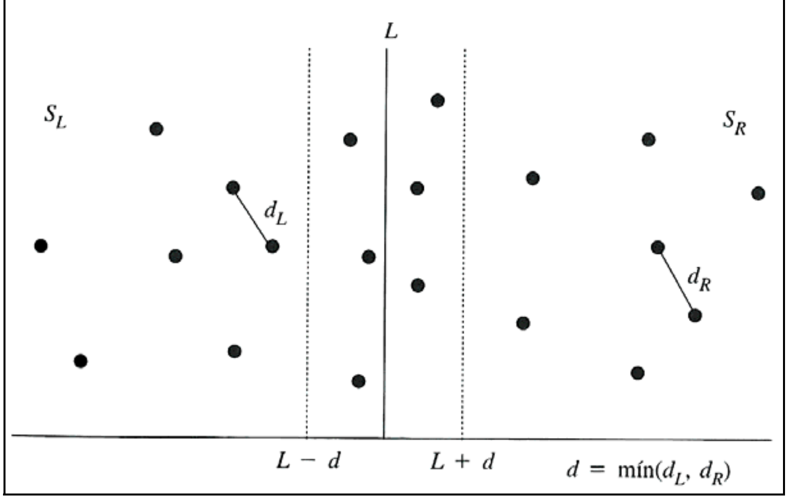


Ilustración 43, representación espacial del problema del par de puntos

Tanto la como la , que aparecen a continuación, se enfocan sobre la región de estudio de los puntos más cercanos.

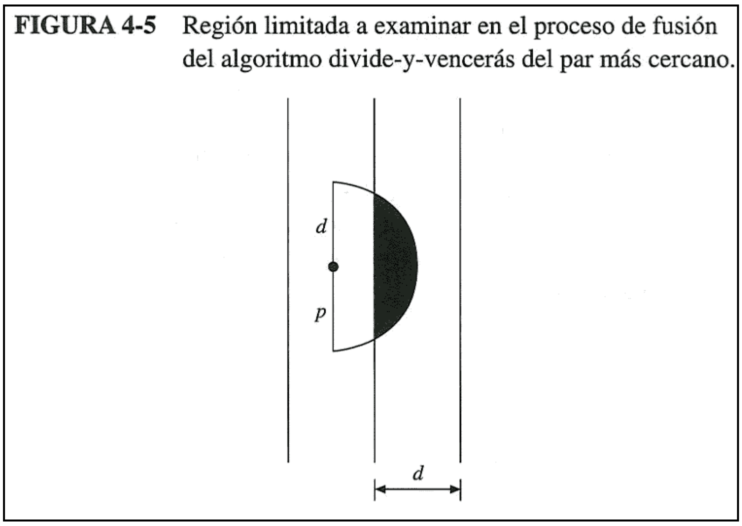


Ilustración 44, representación espacial del problema del par de puntos

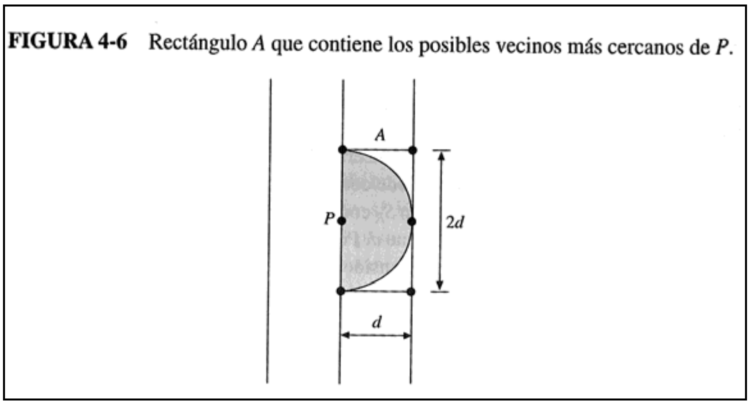


Ilustración 45, representación espacial del problema del par de puntos

7.2 Puntos Dominados por cada Punto del Plano

Tan sólo una obra bibliográfica [11] contiene este problema, dándole el siguiente tipo de ilustración:

- Representación espacial (dependiente del dominio).

Esta representación permite, sin duda, una cómoda exposición del problema que facilita además la comprensión del mismo al permitir ver la relación entre los diferentes puntos. Se muestran dos representaciones mediante la Ilustración 46 y la Ilustración 47 que realiza el autor, donde se remarcan los nodos con un rango máximo, tanto a nivel general como tras una primera división del problema en dos mitades.

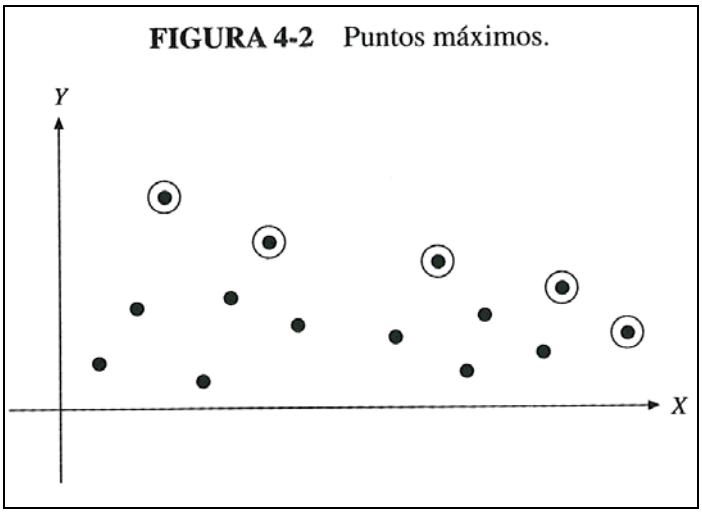


Ilustración 46, representación espacial de puntos máximos

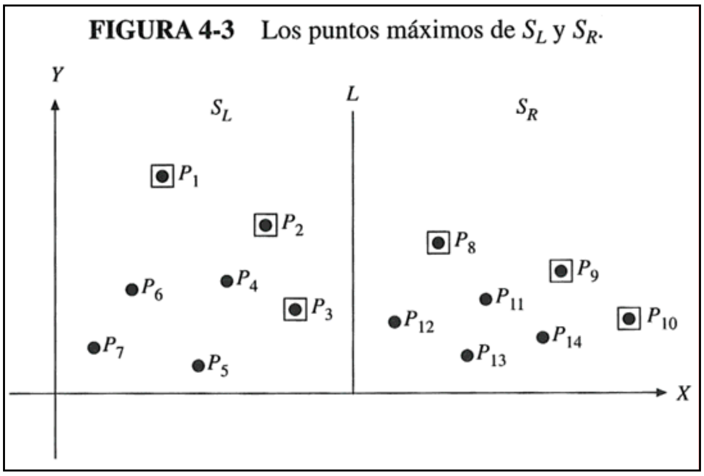


Ilustración 47, representación espacial de puntos máximos con división

7.3 Polígonos Convexos

Únicamente dos obras bibliográficas [11][14] exponen este problema en su interior, haciendo uso de un tipo de ilustración:

- Representación secuencial espacial (dependiente del dominio).

De esta forma, el autor explica paso a paso el modo de construcción, tal y como se puede apreciar en las siguientes representaciones. La Ilustración 48 muestra el proceso de creación de un convex hull y la Ilustración 49 se enfoca en la búsqueda de Graham. Las tres figuras pertenecen a [11].

FIGURA 4-9 Estrategia divide-y-vencerás para construir un convex hull.

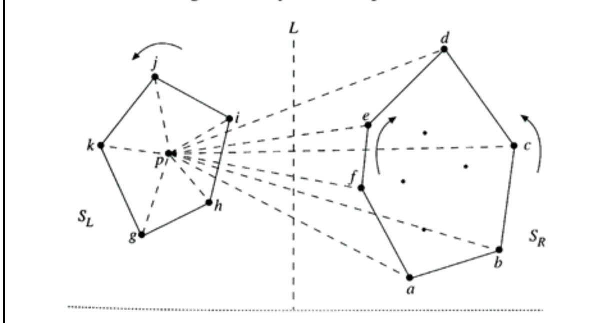


FIGURA 4-11 El convex hull para los puntos en la figura 4-9.

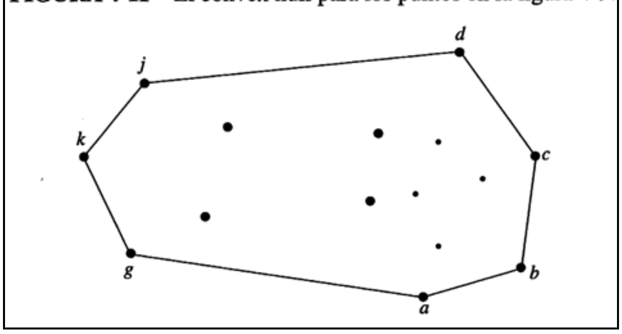


Ilustración 48, rep. geométrica del proceso de creación de un convex hull

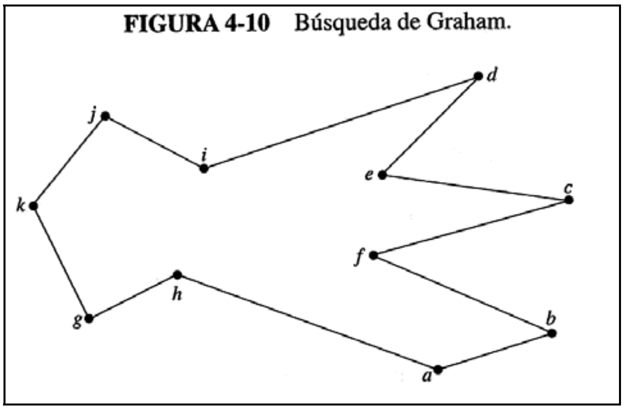


Ilustración 49, representación geométrica de la búsqueda de Graham

7.4 Diagramas de Voronoi

El problema de la generación de diagramas de Voronoi aparece también sólo en una obra bibliográfica [11], que emplea la siguiente representación:

- Representación espacial (dependiente del dominio).

Para la ilustración de este problema, destaca el hecho de que el autor emplea diversos ejemplos y algunas representaciones se acercan a instantes complejos con el fin de explicar el proceso que se realiza. Se muestran a continuación todas las figuras que el autor incluyó en la exposición del problema por medio de la Ilustración 50, la Ilustración 51, la Ilustración 52, la Ilustración 53, la Ilustración 54 y la Ilustración 55.

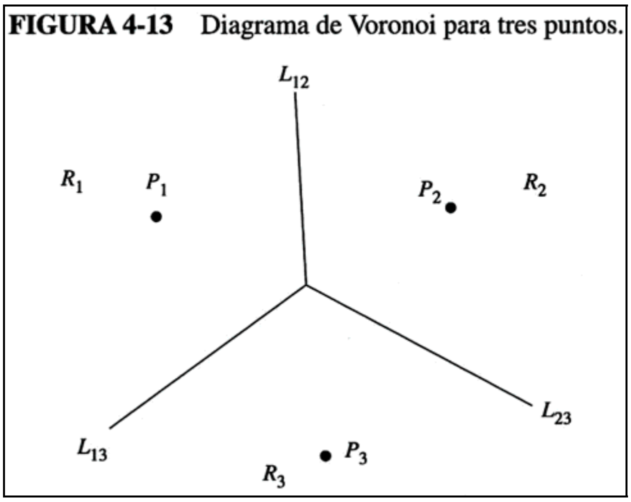


Ilustración 50, diagrama de Voronoi (tres puntos)

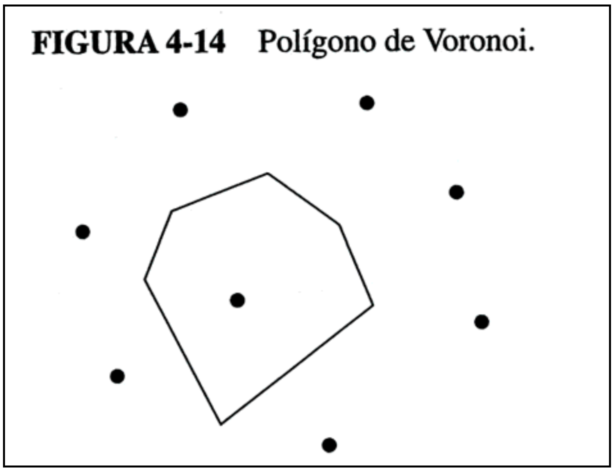


Ilustración 51, polígono de Voronoi

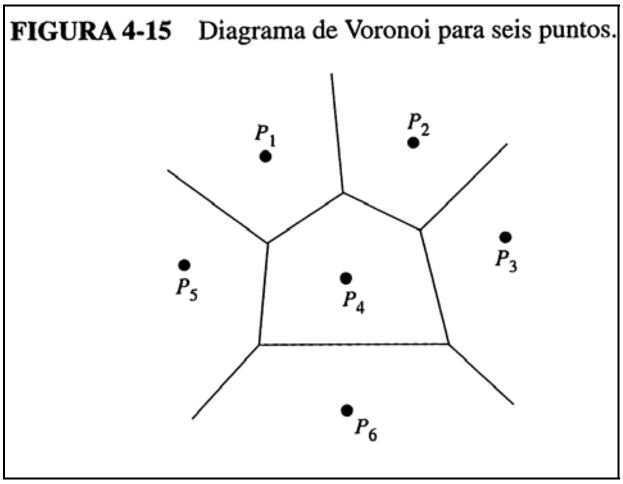


Ilustración 52, diagrama de Voronoi (seis puntos)

FIGURA 4-16 Una triangulación Delaunay.

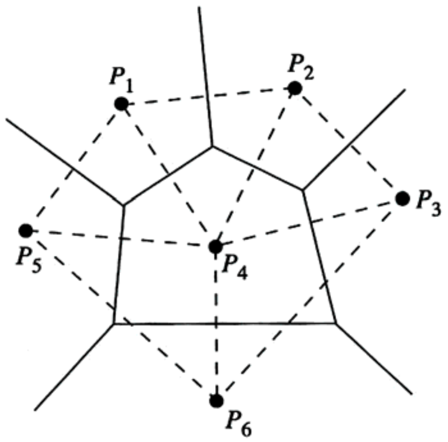


Ilustración 53, triangulación de Delaunay

FIGURA 4-17 Dos diagramas de Voronoi después del paso 2.

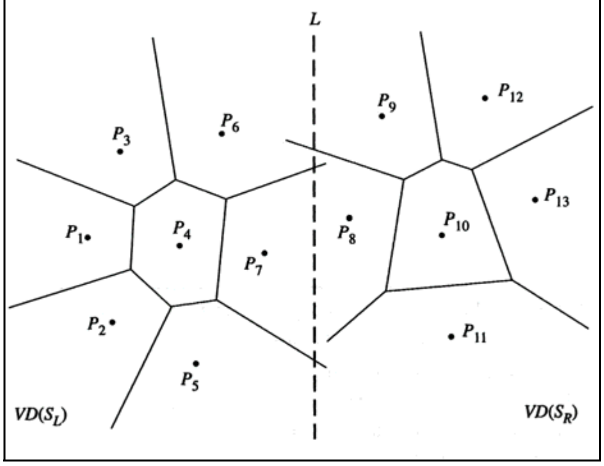


FIGURA 4-18 El hiperplano lineal por partes para el conjunto de puntos que se muestra en la figura 4-17.

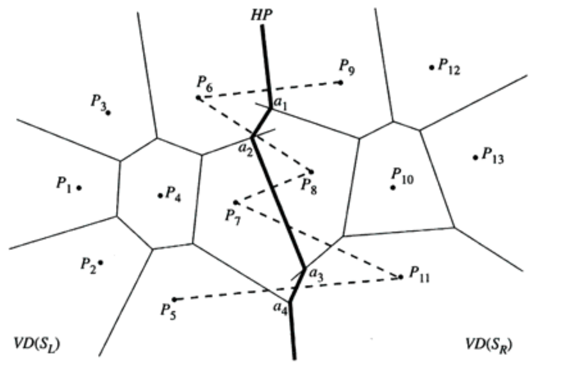


FIGURA 4-19 Diagrama de Voronoi de los puntos en la figura 4-17.

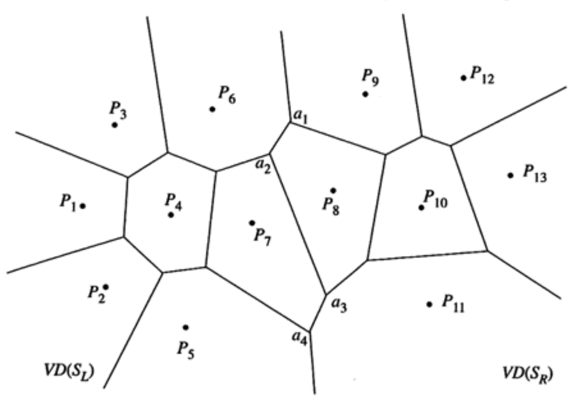


Ilustración 54, proceso de construcción del diagrama de Voronoi

FIGURA 4-20 Otro caso que ilustra la construcción de diagramas de Voronoi.

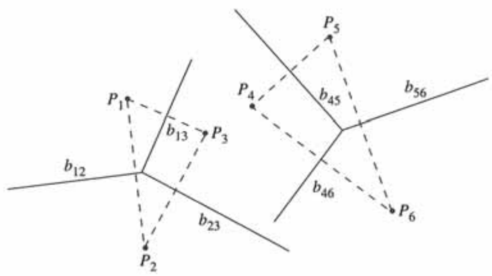


FIGURA 4-21 Paso de fusión en la construcción de un diagrama de Voronoi.

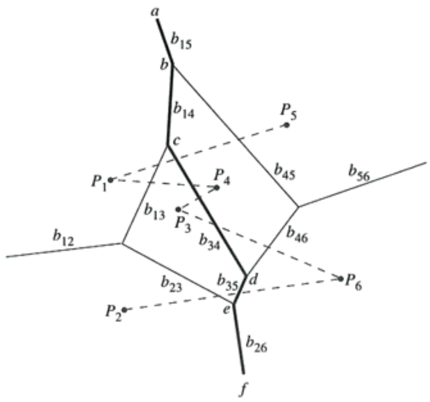


FIGURA 4-22 Diagrama de Voronoi resultante.

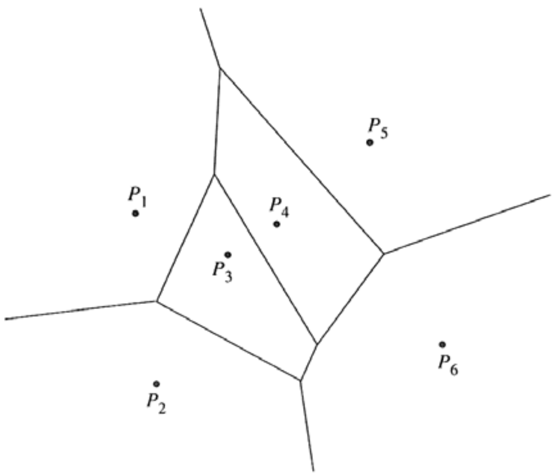


Ilustración 55, proceso de construcción del diagrama de Voronoi

A continuación, la Ilustración 56, la Ilustración 57 y la Ilustración 58 muestran otros procesos relacionados con los diagramas de Voronoi.

FIGURA 4-25 Construcción de un convex hull a partir de un diagrama de Voronoi.

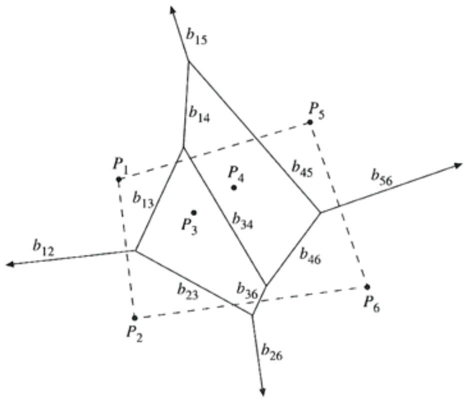


Ilustración 56, construcción de convex hull desde el d. de Voronoi

FIGURA 4-26 Diagrama de Voronoi para un conjunto de puntos en una recta.

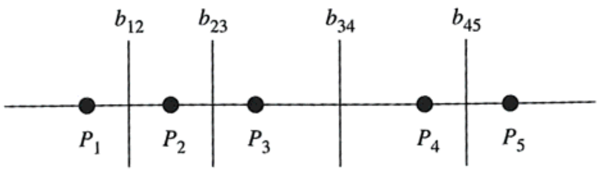


Ilustración 57, diagrama de Voronoi para puntos en línea recta

FIGURA 4-27 Aplicación de los diagramas de Voronoi para resolver el problema euclidiano de búsqueda del vecino más cercano.

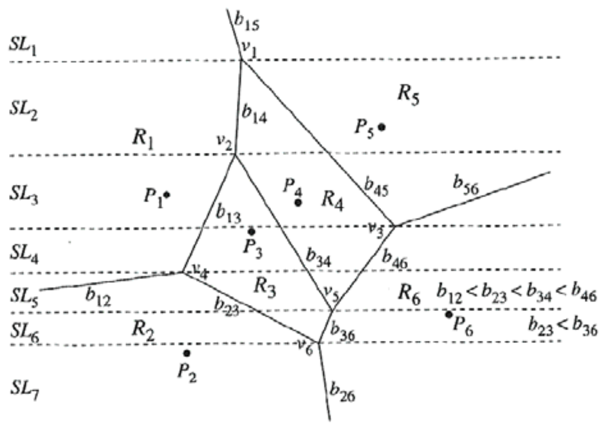


Ilustración 58, búsqueda de vecino más cercano usando el d. Voronoi

8. Otros Problemas

Aquí se recoge un único algoritmo encontrado en la bibliografía utilizada que no entra en ninguna de las categorías anteriores.

Éste es el problema del cálculo del calendario de un campeonato. Éste intenta determinar el repertorio de encuentros que deben disputarse los contrincantes de un campeonato. El algoritmo es representado con una tabla de manera habitual en la bibliografía consultada [4], tal y como se puede ver en la Ilustración 59.

	Jugador					
	Día	1	2	3	4	5
	1	2	1	—	5	4
	2	3	5	1	—	2
(n = 5)	3	4	3	2	1	—
	4	5	—	4	3	1
	5	—	4	5	2	3

	Jugador						
	Día	1	2	3	4	5	6
	1	2	1	6	5	4	3
	2	3	5	1	6	2	4
(n = 6)	3	4	3	2	1	6	5
	4	5	6	4	3	1	2
	5	6	4	5	2	3	1

Figura 7.7. Horarios para cinco y seis jugadores

Ilustración 59, tabla del problema del calendario de competición

9. Tablas de Resumen

Tras la catalogación realizada de representaciones de problemas, clasificados a su vez en diferentes categorías, se pasa a ofrecer un resumen en formato tabular sobre los problemas explorados, aportando los autores que los recogen en sus libros, los tipos de representaciones con que cuentan. Los problemas de distintas categorías aparecen separados por una doble línea.

Tabla 1. Listado de problemas y de sus representaciones

Problema	Bibliografía	Representaciones
Multiplicación de enteros de gran tamaño	[2][3][4] [10][14][15]	• Explicación textual. • Esquema de la división de los números • Representación matemática
Multiplicación de matrices	[2][3][4][9] [13][14][15]	• Explicación textual. • Representación matemática
Exponenciación	[3][4]	• Explicación textual.
Cálculo de factoriales	[6]	• Pila de control.
Cálculo de la serie de números de Fibonacci	[3]	• Explicación textual.
Convolución / T. Fourier	[10][11]	• Explicación textual.

Búsqueda binaria	[2][3][4][5] [14][15]	• Explicación textual. • Secuencia de los índices que limitan la parte tratada del vector. • Árbol.
Máximo de un vector	[11]	• Árbol completo secuencial.
Mínimo y máximo de un vector	[13][15]	• Árbol con símbolos. • Explicación textual
Selección	[2][3][4] [13]	• Representación textual. • Secuencia del vector, con realzado de partes importantes.
Mediana de vector mezcla de dos vectores ordenados	[12]	• Esquema de los vectores de entrada y del valor resultante
Elemento mayoritario de un vector	[3]	• Representación textual.
Mergesort	[2][3][4][9] [10][13][14]	• Representación textual. • Árbol de activación, con flechas y numeración secuencial. • Secuencia de la tarea de combinación. • Secuencia de la partición y unión del vector. • Secuencia de la división del vector.
Quicksort	[2][3][4][5] [13][14][15]	• Representación textual. • Secuencia del vector marcando las divisiones y la posición del pivote. • Esquema con el vector dividido.
Coloreado de tablero defectuoso	[9][13]	• Tablero con uno o más triominos. • Tablero sobre el que se dibujan las divisiones que se realizan.
Intercambio de partes de un vector	[3]	• Esquema con sombreado en las zonas de movimiento • Secuencia con índices apuntando a los límites de las zonas manejables.
Par de puntos más cercanos entre sí en un conjunto	[9][13][14]	• Representación espacial.
Puntos dominados por cada punto del plano	[11]	• Representación espacial.
Polígonos convexos	[11][14]	• Representación secuencial espacial.
Diagramas de Voronoi	[11]	• Representación espacial.
Competición	[4]	• Tabla.

Podemos detallar más el análisis si nos limitamos a las ilustraciones, como se muestra en la tabla siguiente.

Tabla 2. Listado de ilustraciones y del tipo de representación que contienen

Ilustración	Problema	Representaciones
1.	Multiplicación de enteros	Representación dependiente del dominio (explicación) Etiquetas alfabéticas, líneas (rangos) y fórmulas
2.	Multiplicación de enteros	Representación dependiente del dominio (explicación) Etiquetas alfabéticas y fórmulas
3.	Multiplicación de enteros	Figura doble, con 2 secuencias de estados Representación dependiente del dominio (multiplicación convencional: cómputo)
4.	Multiplicación de enteros	Representación dependiente del dominio (fórmula con notación matricial: explicación) Etiquetas alfabéticas y operadores
5.	Factorial	Secuencia de estados (pila de control: cómputo) Etiquetas de identificadores y sentencias
6.	Búsqueda binaria	Figura compuesta (cómputo): • Estado de vector • Secuencia de estados, representados mediante la posición de índices y resultado a la derecha Comprobaciones a la derecha
7.	Búsqueda binaria	Árbol binario de búsqueda (ejemplo) Nodos circulares con índice interno
8.	Búsqueda binaria	Vector (explicación) Etiqueta para resultado y cada celda, llaves y fórmulas
9.	Máximo de un vector	Árbol de recursión (cómputo) Nodos rectangulares, con subvector Arcos para valores calculados
10.	Mínimo y máximo de un vector	Figura doble, con 2 árboles de recursión (partición de datos y combinación de resultados: explicación) Nodos: circulares o sin marco Etiquetas alfabéticas
11.	Mínimo y máximo de un vector	Árbol de recursión (explicación) Nodos circulares Etiquetas alfabéticas y sombreado
12.	k-ésimo valor más pequeño	Secuencia de vectores (cómputo) Celdas del subvector no seleccionade sin valor (punto)
13.	k-ésimo valor más pequeño	Representación dependiente del dominio (plano euclídeo: explicación) Etiquetas alfabéticas y texto Puntos, áreas y flechas
14.	k-ésimo valor más pequeño	Vector subdividido por ubicación (explicación) Zonas sombreadas para valores mayores que la mediana, uso selectivo de etiquetas, flechas
15.	k-ésimo valor más pequeño	Vector subdividido por ubicación y separación de líneas (explicación) Zonas separadas por líneas, posiciones etiquetadas

16.	Mediana de vector mezcla de dos vectores ordenados	Vectores (entrada y salida) con celda objetivo sombreada (explicación)
17.	Ordenación por mezcla	Secuencia de estados (vectores: cómputo inductivo) Estados de entrada y de resultado
18.	Ordenación por mezcla	Secuencia de estados (vectores: cómputo) Agrupamiento de los de un mismo nivel de recursión Etiqueta (operación) a la izquierda
19.	Ordenación por mezcla	Secuencia de estados (vectores: cómputo), Agrupamiento de los de un mismo nivel de recursión, espacios de separación entre subvectores Todos los vectores de entrada y de resultado
20.	Ordenación por mezcla	Árbol de recursión (cómputo) Nodos rectangulares, con subvector Imágenes especulares superior e inferior para flujo activo y pasivo
21.	Ordenación por mezcla	Árbol de recursión (explicación) Nodos circulares, con etiquetas de tamaño Texto a la derecha de cada nivel (rendimiento)
22.	Ordenación por mezcla	Árbol de recursión (cómputo) Nodos rectangulares, con subvectores de entrada y salida (sombreados) Fechas hacia arriba y abajo (flujo de ejecución)
23.	Ordenación por mezcla	Árbol de recursión (cómputo) Nodos rectangulares, flechas hacia arriba indicando dónde se produce la acción de mezcla
24.	Ordenación por mezcla: mezcla	Secuencia de estados (vectores: cómputo) Flechas señalan la posición apuntada por índices
25.	Ordenación por mezcla	Árbol de partición (cómputo)
26.	Ordenación rápida: partición	Secuencia de estados (vectores: cómputo) Líneas separadoras de celdas (sencillas y dobles)
27.	Ordenación rápida: partición	Secuencia de estados (vectores: cómputo) Etiquetas posicionadas de índices Barras separadoras y sombreado
28.	Ordenación rápida: partición	Figura doble, con 2 secuencias de vector (explicación de 2 casos) Etiquetas posicionadas de índices y condiciones Barras separadoras, sombreado, llaves y flechas
29.	Ordenación rápida: partición	Figura doble, con 2 vectores (explicación de 2 casos) Etiquetas posicionadas de índices y condiciones Barras separadoras y sombreado
30.	Ordenación rápida: partición	Figura triple, con 3 vectores (explicación de 3 casos) Etiquetas posicionadas de índices y condiciones Barras separadoras y flechas
31.	Ordenación rápida: partición	Figura con un vector donde se van superponiendo secuencialmente los índices en cada paso (cómputo)

		Etiquetas para índices y posiciones del vector
32.	Ordenación rápida: partición	Árbol de recursión (cómputo) Nodos rectangulares, dos partes: área que se maneja y posición de división
33.	Ordenación rápida: partición	Conjunto no ordenado con las posiciones del vector, los grupos se ordenan a medida que lo hace el vector (cómputo) Texto ubicado en proceso, flechas
34.	Embaldosar un tablero defectuoso	Figura doble, con representaciones dependientes del dominio (partición de un tablero): • Explicación: fórmulas internas • Un paso de cómputo: áreas y sombreados
35.	Embaldosar un tablero defectuoso	Secuencia de 3 representaciones dependientes del dominio (ejemplo) Áreas y sombreados
36.	Embaldosar un tablero defectuoso	Secuencia de 5 representaciones dependientes del dominio (ejemplo) Áreas y sombreados
37.	Intercambio de partes de un vector	Figura triple, con vectores: • Problema: estado inicial, etiquetas alfabéticas para las celdas • Explicación: etiquetas de variables y texto, flechas, sombreado y llaves (rangos) • Cómputo: estado tras un paso de ejecución, mismo formato que la 1ª parte
38.	Intercambio de partes de un vector	Secuencia de vectores Etiqueta de cada estado (llamada) Etiquetas alfabéticas para las celdas Flechas para índices Llaves para rangos
39.	Par de puntos más cercanos	Representación dependiente del dominio (plano euclídeo: explicación) Distinción de áreas mediante líneas y sombreado Etiquetas alfabéticas (áreas y distancia)
40.	Par de puntos más cercanos	Representación dependiente del dominio (explicación) Uso de etiquetas, puntos, áreas, líneas y sombreado
41.	Par de puntos más cercanos	Representación dependiente del dominio (ejemplo)
42.	Par de puntos más cercanos	Representación dependiente del dominio (cómputo por divide y vencerás) Etiquetas y líneas (distancias)
43.	Par de puntos más cercanos	Figura doble de: • Explicación de cómputo por divide y vencerás • ¿Propiedad? Etiquetas, líneas y flechas
44.	Puntos maximales	Representación dependiente del dominio (plano euclídeo: problema con solución) Resaltado (marcos)
45.	Puntos maximales	Representación dependiente del dominio (plano

		euclídeo: cómputo por divide y vencerás) Etiquetas, líneas y marcos
46.	Polígono convexo	Representación dependiente del dominio (plano euclídeo: cómputo por divide y vencerás) Etiquetas, líneas y flechas
47.	Polígono convexo	Representación dependiente del dominio (plano euclídeo: problema) Etiquetas
48.	Polígono convexo	Representación dependiente del dominio (plano euclídeo: solución) Etiquetas
49.	Diagrama de Voronoi	Representación dependiente del dominio (plano euclídeo: explicación) Etiquetas y líneas
50.	Diagrama de Voronoi	Representación dependiente del dominio (plano euclídeo: explicación) Líneas
51.	Diagrama de Voronoi	Representación dependiente del dominio (plano euclídeo: problema) Etiquetas y líneas
52.	Diagrama de Voronoi	Representación dependiente del dominio (plano euclídeo: cómputo) Etiquetas y líneas
53.	Diagrama de Voronoi	Representación dependiente del dominio (plano euclídeo: cómputo) Etiquetas y líneas
54.	Diagrama de Voronoi	Representación dependiente del dominio (plano euclídeo: cómputo) Uso de etiquetas y líneas
55.	Calendario de competición	Figura doble, con 2 tablas (soluciones) Etiqueta de tamaño del problema (tabla) En ambas dimensiones, etiqueta de variable e índices

10. Conclusiones

Hemos realizado un estudio exhaustivo de las ilustraciones contenidas en libros de texto de algoritmos sobre la técnica de divide y vencerás. El análisis de sus características se deja para un trabajo posterior.

Agradecimientos. Este trabajo se ha financiado con el proyecto TIN2011-29542-C02-01 del Ministerio de Innovación y Ciencia.

Referencias

[1] Allen, M. (1995): Estructuras de datos y algoritmos. Addison-Wesley Iberoamericana (Cap. 10.2).
[2] Alsuwaiyel, M. H. (1999): Algorithms, design techniques and analysis. World Scientific (Cap. 6).
[3] Brassard, G., Bratley, P. (1996): Algorítmica, concepción y análisis. Masson (Cap. 4).
[4] Brassard, G., Bratley, P. (1997): Fundamentos de algoritmia. Prentice Hall (Cap. 7)
[5] Cormen, T.H., Leiserson, C.E., Rivest, R.L. (2005): Introduction to algorithms. The MIT Press.
[6] De Giusti, A. (2001): Algoritmos, datos y programas con aplicaciones en Pascal, Delphi y Visual Da Vinci. Prentice Hall (Cap. 7).
[7] Di Batista, P., Eades, G., Tamassia, T., Toillis, I. (1999): Graph Drawing: Algorithms for the Visualization of Graphs. Prentice-Hall (Cap. 3).
[8] Gonzalo A. (1997): Esquemas algorítmicos: enfoque metodológico y problemas resueltos. Universidad Nacional de Educación a Distancia (Cap. 3).
[9] Johnsonbaugh, R., Schaefer, M. (2004): Algorithms (internacional edition). Pearson Education (Cap. 5).
[10] Kleinberg, J., Tardos, É. (2006): Algorithm design. Pearson Addison-Wesley (Cap. 5).
[11] Lee, R.C.T., Tseng, S.S., Chang R.C., Tsai, Y.T. (2007): Introducción al diseño y análisis de algoritmos, un enfoque estratégico. Mc Graw Hill (Cap. 4).
[12] Martí N. (2004): Estructuras de datos y métodos algorítmicos ejercicios resueltos. Pearson Prentice Hall (Cap. 11).
[13] Sahni, S. (2005): Data structures, Algorithms and Applications in Java. Silicon Press (Cap. 19).
[14] Levitin, A. (2003): The Design of Analysis of Algorithms. Wesley (Cap. 4).
[15] Parberry, I (2002): Problems on Algorithms. Prentice-Hall. (Cap. 7)