Manuel Arturo Chinchilla Sánchez - 2013009344

Resumen #12

Chapter 5.2: Divide and Conquer.

Divide y Conquista.

Implica la resolución de un problema computacional particular, dividiéndolo en uno o más subproblemas de tamaño más pequeño, la resolución recursiva de cada subproblema, y luego la "fusión" de las soluciones a los subproblemas para producir una solución al problema original.

Podemos modelar el enfoque de divide y vencerás utilizando un parámetro *n* para indicar el tamaño del problema original, y dejar que *S(n)* denote este problema. Solucionamos el problema *S(n)* mediante la resolución de un conjunto *k* de subproblemas $S(n_1)$, $S(n_2)$, ..., $S(n_k)$, donde $n_i < n$ para i = 1, ..., k, y a continuación, la fusión de las soluciones a estos subproblemas.

Ecuaciones de Recurrencia de Divide y Conquista.

Para analizar el tiempo de ejecución de un algoritmo de divide y vencerás utilizamos una ecuación de recurrencia. Es decir, dejamos que una función T(n) denote el tiempo de ejecución del algoritmo en una entrada de tamaño n, y caracterizar T(n) utilizando una ecuación que relacione T(n) a los valores de la función de T para los tamaños de problema menor que n. En el caso del algoritmo de fusión, obtenemos la ecuación de recurrencia:

$$T(n) = \begin{cases} b \ \boldsymbol{si} \ n < 2 \\ 2T(n/2) + bn \ \boldsymbol{si} \ n \geq 2 \end{cases}$$

Para alguna constante b> 0, tomando la suposición simplificadora de que *n* es una potencia de 2. De hecho tomamos el supuesto simplificador de que *n* es una potencia adecuada, de modo que podamos evitar el uso de funciones de parte entera. Cada declaración asintótica que tomamos sobre ecuaciones de recurrencia seguirá siendo cierta, incluso si relajamos esta hipótesis, pero justificando este hecho implica formalmente pruebas largas y aburridas. Como se observó anteriormente, podemos demostrar que T(n) es O (n log n), en este caso. En general, sin embargo, vamos a conseguir posiblemente una ecuación de recurrencia que es más difícil de resolver que esta. Por lo tanto, es útil para desarrollar algunos aspectos generales de la solución de los tipos de ecuaciones de recurrencia que surgen en el análisis de divide y vencerás.

Multiplicación de Enteros.

Consideramos el problema de multiplicar enteros grandes, que están representados por una número largo de bits que no pueden ser manejados directamente por la unidad de aritmética de un solo procesador. La multiplicación de grandes enteros tiene aplicaciones en la seguridad de datos, donde los grandes enteros son usados en esquemas de encriptación.

Dados dos enteros grandes I y J representados con n bits cada uno, podemos fácilmente calcular $I + J$ y $I - J$ en un tiempo O(n). Calcular eficientemente el producto $I \cdot J$ usando el algoritmo común de la escuela requiere un tiempo $O(n^2)$.

Ahora asumimos que n es una potencia de 2 (si este no es el caso, podemos rellenar I y J con ceros). Podemos para ello dividir las representaciones de bits de I y J en mitades, con una mitad representando el orden más alto de bits y la otra mitad representando el orden más pequeño de bits. En particular, si partimos I en I_h y I_l y J en J_h y J_l, entonces:

$$I = I_h 2^{n/2} + I_l$$

$$J = J_h 2^{n/2} + J_h$$