a) Demostración de que la altura del AVL es O(log n) (Informal)

La altura de un Árbol AVL con 'n' nodos es O(log n) debido a su estricta condición de equilibrio. Esta condición establece que la diferencia de alturas entre los subárboles de cualquier nodo es como máximo 1 (Factor de Equilibrio entre -1, 0, y 1).

Para entenderlo, se analiza el árbol AVL con el mínimo número de nodos (n(h)) para una altura h. Este número sigue una secuencia recurrente similar a la de los números de Fibonacci: $n(h) = 1 + n(h-1) + n(h-2)$. Dado que el número de nodos 'n' crece exponencialmente con la altura 'h', su inversa (la altura 'h') debe crecer logarítmicamente con 'n'. Por lo tanto, la altura del AVL está acotada superiormente por O(log n).

b) Garantía de operaciones en O(log n)

El rendimiento logarítmico de las operaciones se garantiza porque la altura del árbol siempre es O(log n), lo que limita la longitud del camino más largo a recorrer.

Búsqueda (Search): La operación de búsqueda consiste en seguir un camino desde la raíz hasta el nodo. El costo en el peor caso está limitado por la altura, que es O(log n).

Inserción (Insert): La inserción incluye la búsqueda del lugar (O(log n)) y el posible reequilibrio. El reequilibrio se realiza con un máximo de una rotación simple o doble, una operación de costo constante O(1). El costo total sigue siendo O(log n).

Eliminación (Delete): La eliminación también se realiza en tiempo O(log n). Aunque la eliminación puede requerir más de una rotación (una por cada nivel en el camino de vuelta a la raíz) para reestablecer el balance, el número de rotaciones está limitado por la altura, por lo que el costo total de reequilibrio es O(log n).

c) Comparación conceptual (AVL vs ABB vs Rojinegros)

Árbol Binario de Búsqueda (ABB) sin Balance: No tiene reglas de balanceo. Su principal problema es que en el peor caso (por ejemplo, al insertar datos ordenados), el árbol se degenera en una lista enlazada, y la altura se convierte en O(n), llevando a un rendimiento pobre O(n) en todas las operaciones.

Árbol AVL: Es un árbol de búsqueda binario de auto-balanceo con un balance muy estricto. Esto resulta en la altura más baja posible, lo que lo hace muy rápido para operaciones de búsqueda (lectura). Sin embargo, el estricto balanceo puede requerir más operaciones de rotación durante las inserciones y eliminaciones comparado con los Rojinegros.

Árbol Rojinegro (Red-Black Tree - RBT): Es otro árbol de auto-balanceo con una política de balanceo más relajada que el AVL. Su altura también es O(log n). Aunque es marginalmente menos eficiente en búsquedas que un AVL (es un poco más alto), es más eficiente en operaciones de inserción y eliminación porque requiere menos rotaciones para mantener su balance. Por esta razón, suele ser el árbol de balanceo preferido en las librerías de lenguajes de programación.