OBSERVACIONES DE LA PRÁCTICA

Mateo Sánchez Zapata 202321354
Santiago Garzon Garcia 202512373
Santiago Ismael Escobar 202516956

Preguntas de análisis

1. ¿Qué diferencia existe entre las alturas de los dos árboles (BST y RBT)?
   La altura del árbol binario de búsqueda (BST) depende directamente del orden en que se insertan los elementos. Si los datos se insertan de forma ordenada (ascendente o descendente), el BST se degrada y su altura puede llegar a ser n, convirtiéndose en una lista lineal.
   Por otro lado, el árbol rojo-negro (RBT) mantiene su balanceo automático mediante rotaciones y recoloreos en cada inserción o eliminación, garantizando que su altura máxima sea aproximadamente $2 \cdot \log_2(n + 1)$.
   En resumen:

   - El BST puede tener una altura muy grande (incluso n).
   - El RBT mantiene una altura mucho menor y más equilibrada.

2. ¿Percibe alguna diferencia entre la ejecución de los dos árboles (RBT y BST)? ¿Por qué pasa esto?

   Sí, hay una diferencia notable.
   Las operaciones de inserción, búsqueda y eliminación en un BST dependen de su forma: si está desbalanceado, esas operaciones pueden tardar más porque deben recorrer más niveles.
   En cambio, el RBT ajusta su estructura con rotaciones y recoloreos para mantener el balance, lo que hace que las operaciones sean más uniformes y rápidas en promedio.
   Por eso, en la ejecución, el RBT suele ser más constante y eficiente, especialmente con grandes volúmenes de datos o inserciones desordenadas.

3. ¿Existe alguna diferencia de complejidad entre los dos árboles (RBT y BST)? Justifique su respuesta.
   Sí, hay diferencia en el peor caso.

   BST:

   Mejor caso: O(log n) (si está balanceado).

   Peor caso: O(n) (si está desbalanceado).

   RBT:
   Siempre garantiza O(log n) en búsqueda, inserción y eliminación gracias a su balanceo.

En conclusión, el RBT asegura una eficiencia logarítmica, mientras que el BST solo la alcanza si está balanceado manualmente.

4. ¿Existe alguna manera de cargar los datos en un árbol RBT de tal forma que su funcionamiento mejore? Si es así, mencione cuál.

   Aunque el RBT ya se balancea automáticamente, su rendimiento puede mejorar dependiendo de cómo se cargan los datos.

   Algunas estrategias útiles son:

   Cargar los datos en lotes (bulk load): insertar en grupos grandes reduce las rotaciones individuales.

   Evitar inserciones totalmente ordenadas: aunque el RBT lo corrige, esto genera más recoloreos y rotaciones al inicio.

   Preprocesar los datos de forma aleatoria: ayuda a distribuir las claves mejor desde el principio.

   En resumen: no se puede “mejorar” su algoritmo de balanceo, pero sí minimizar los costos de reestructuración cargando los datos en un orden menos sesgado o por lotes.