Digital search structure

Ch. 12

Ch. 12: Digital search structure

- **Digital Search Tree**

Reduce number of comparisons

- **Trie**

Reduce required space for index

- **Compressed trie**

Reduce required space for data

- **Patricia**

1. Digital search tree

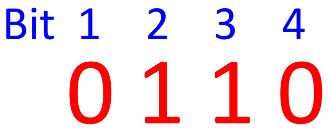
- A binary tree. Each node contains one data pair.

- Keys are binary bit strings, indicating the path of search.
  - For previously introduced structures, key values are meaningful, and we do compare based on key values.
  - Fixed length: 0110, 0010, 1010, 1011.
  - Variable length: 01, 00, 101, 1011.

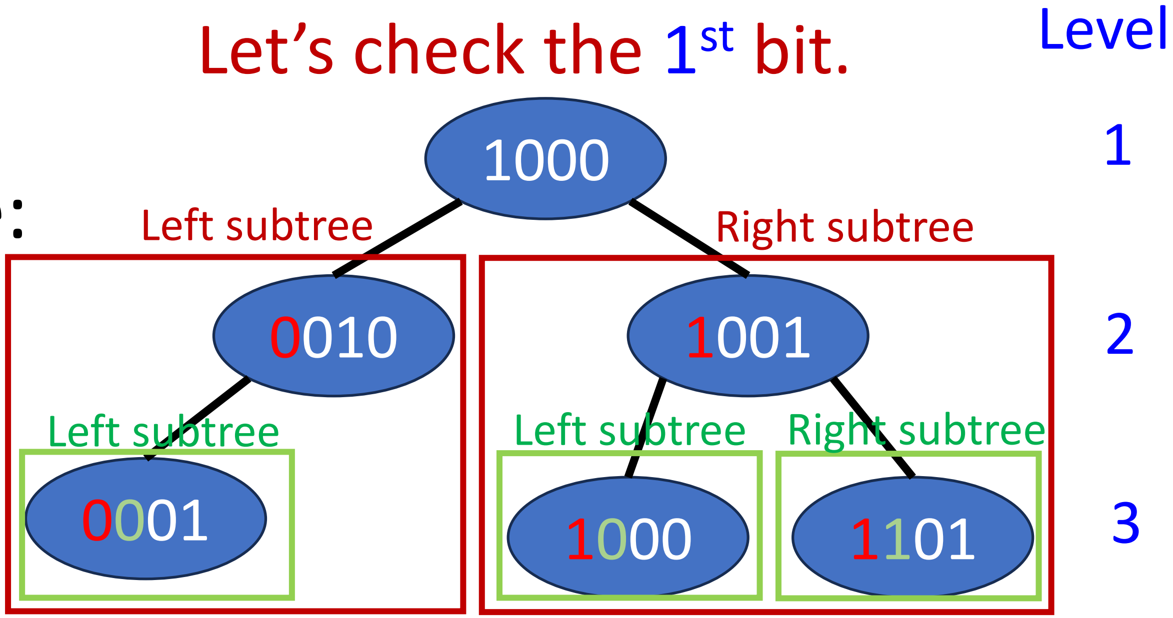
- Applications
  - Networks, e.g., hardware IP routing

Digital search tree

- Assume that the key is fixed length.
- Number the bit from left to right.



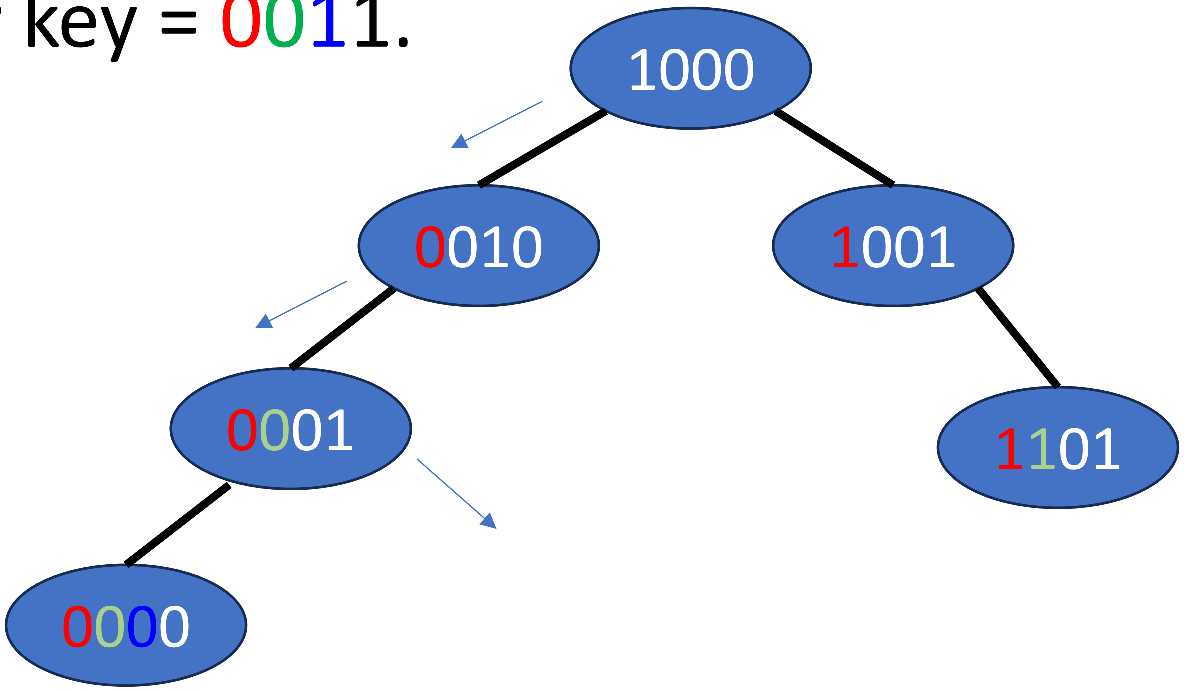
- For a node at level *i*:
  - Keys of pairs in its left subtree: the *i*-th bit = 0.
  - Keys of pairs in its right subtree: the *i*-th bit = 1.



Let's check the 2nd bit.

Operation: Search

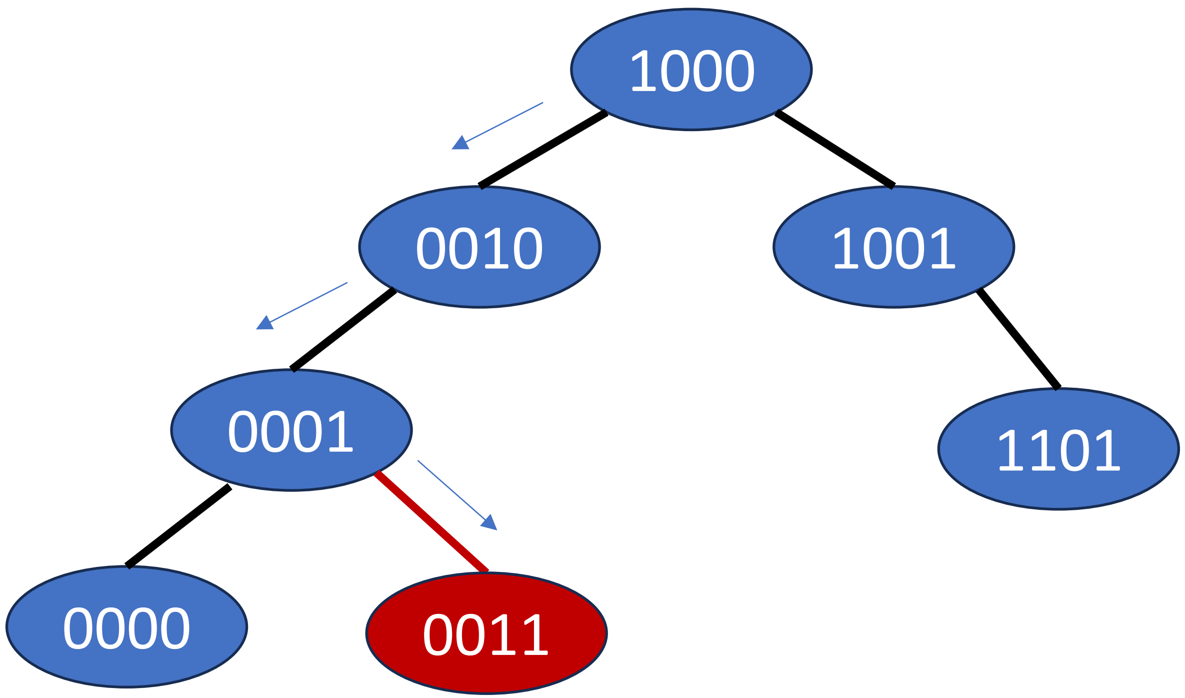
- Search for key = 0011.



- Compare with the key in the root (1000). → different
- 1st bit is 0. Compare with the key in the left child (0010). → different
- 2nd bit is 0. Compare with the key in the left child (0001). → different
- 3rd bit is 1. Compare with the key in the right child. → No right child.
- 0011 is not in the search tree.

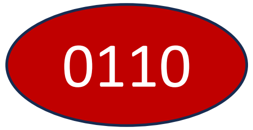
Operation: Insertion

- Insert a data pair whose key = 0011.

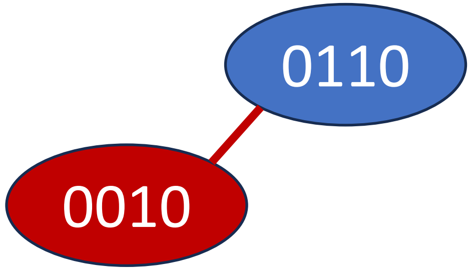


Operation: Insertion

- Start with an empty digital search tree.
- Insert a pair whose key is 0110.



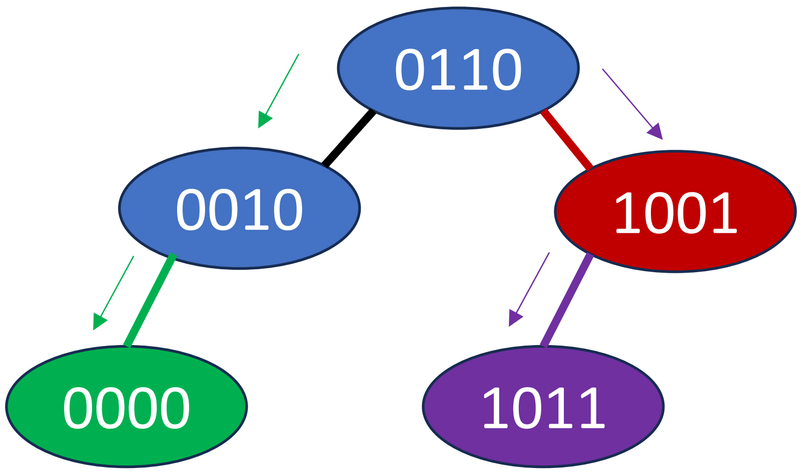
- Then, insert a pair whose key is 0010.
  - The first bit is 0. Move to the left child.



Operation: Insertion

- Then, insert a pair whose key is 1001.
  - The first bit is 1. Move to the right child.

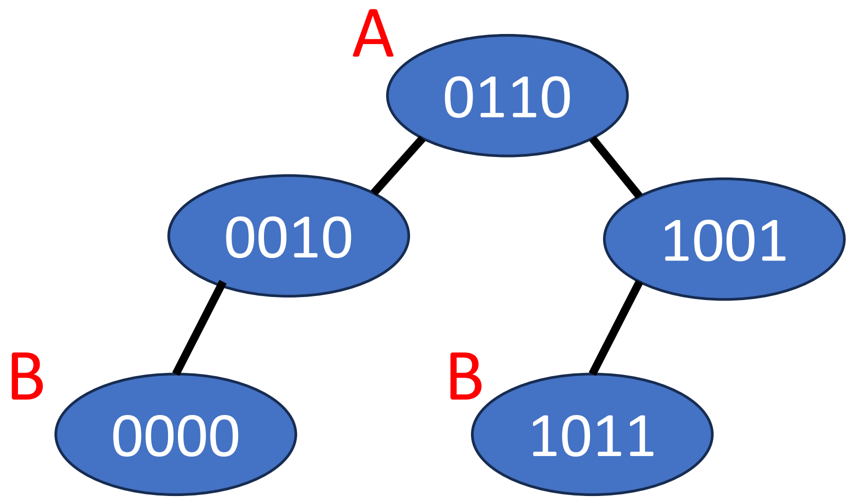
- Then, insert a pair whose key is 1011.
  - The first bit is 1. Move to the right child.
  - The second bit is 0. Move to the left child.

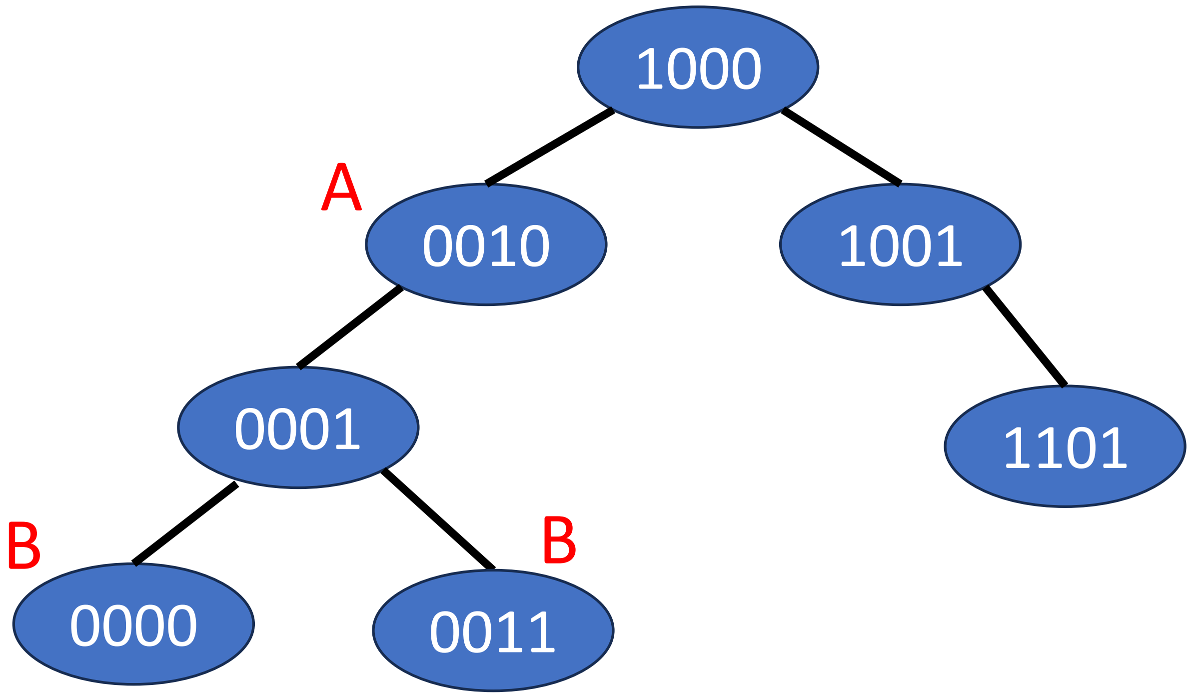


- Then, insert a pair whose key is 0000.
  - The first bit is 0. Move to the left child.
  - The second bit is 0. Move to the left child.

Operation: Deletion

- Delete node A.
  - Find any leaf B in the subtree rooted at A.
  - Then replace A with B.





Exercise

- Q5: Given the following keys 0000, 0011, 0001, 0010, 1000, 1001, 1100, 0110. Following this order, please insert the keys into an empty digital search tree. What will be the keys in the nodes of the 3rd level of the resultant tree?
- Q6: (Continue Q5) When deleting the node whose key is 0011, what is(are) the key(s) to replace key 0011?

Please reply your answers of Q5-Q6 via the following link:

https://forms.gle/NyCVWyzZ6eXwVdYg9

Group members: 2~4 people

Discussion

- Complexity of each operation is O(#bits in a key).
- #key comparisons = O(height).
  - Starting from root, we compare the key with each of the node along the path.
  - The height is at most 1 + #bits in a key.
- Expensive when keys are very long.
  - If keys are computed by SHA-1 hash function with 160 bits, the cost of a key comparison is high.

Can we reduce the number of key comparisons done during a search?

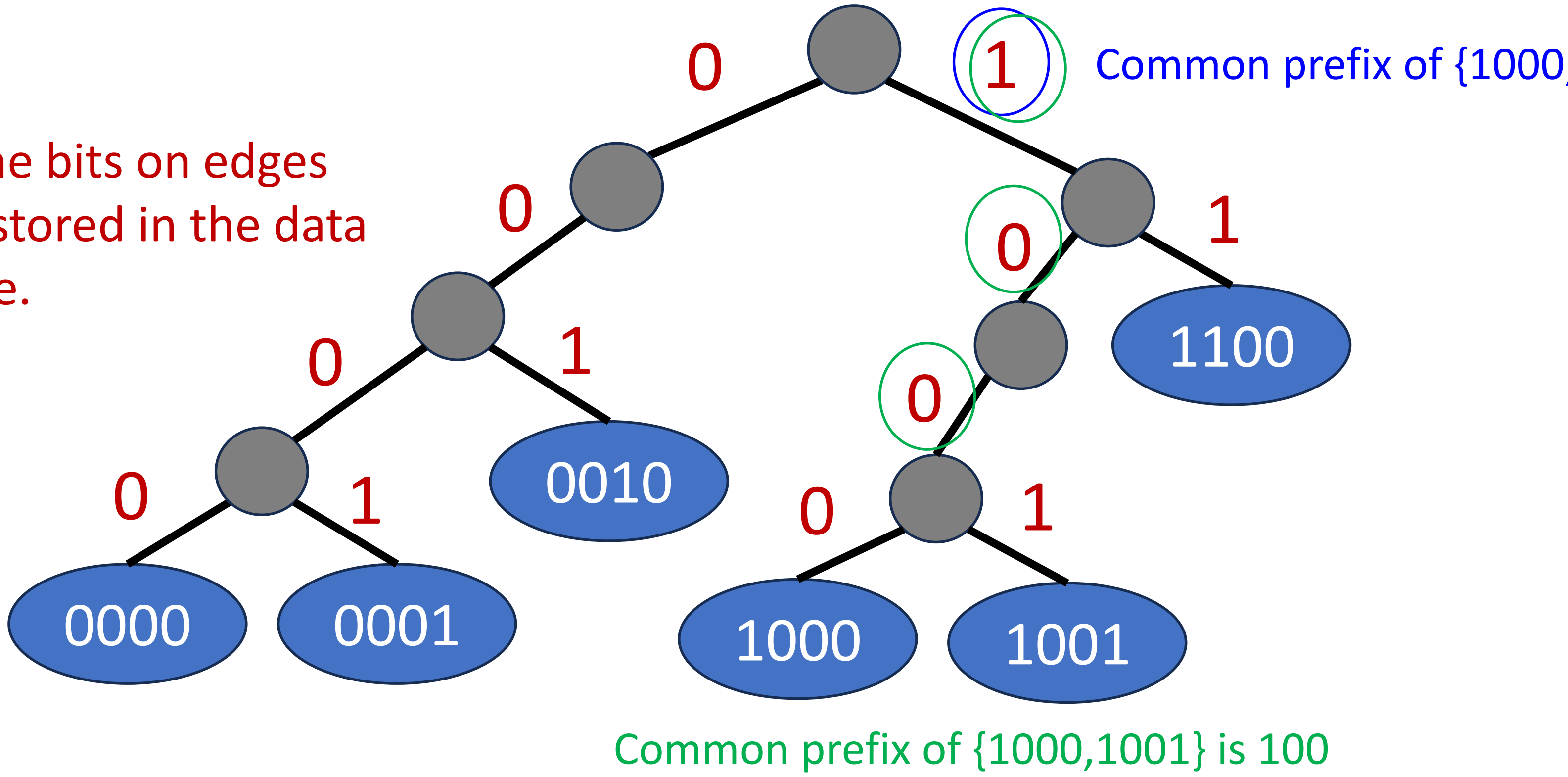
2. Binary Trie

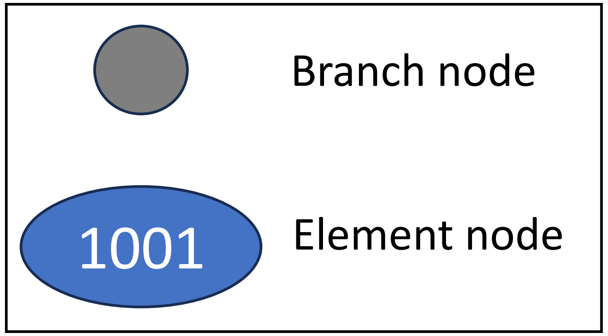
- Information Re**trie**val.
- At most one key comparison per operation.
- Fixed length keys.
- Two types of nodes:
  - Branch nodes
    - Left and right child pointer
    - No data field
  - Element nodes
    - No child pointer
    - Data field to hold dictionary data pair

Example

- Use bits to guide the direction to traverse top-down the tree.
- At most one key comparison.
  - The traversal does not introduce comparisons.
  - When finally reaching an element node, comparing with its key.

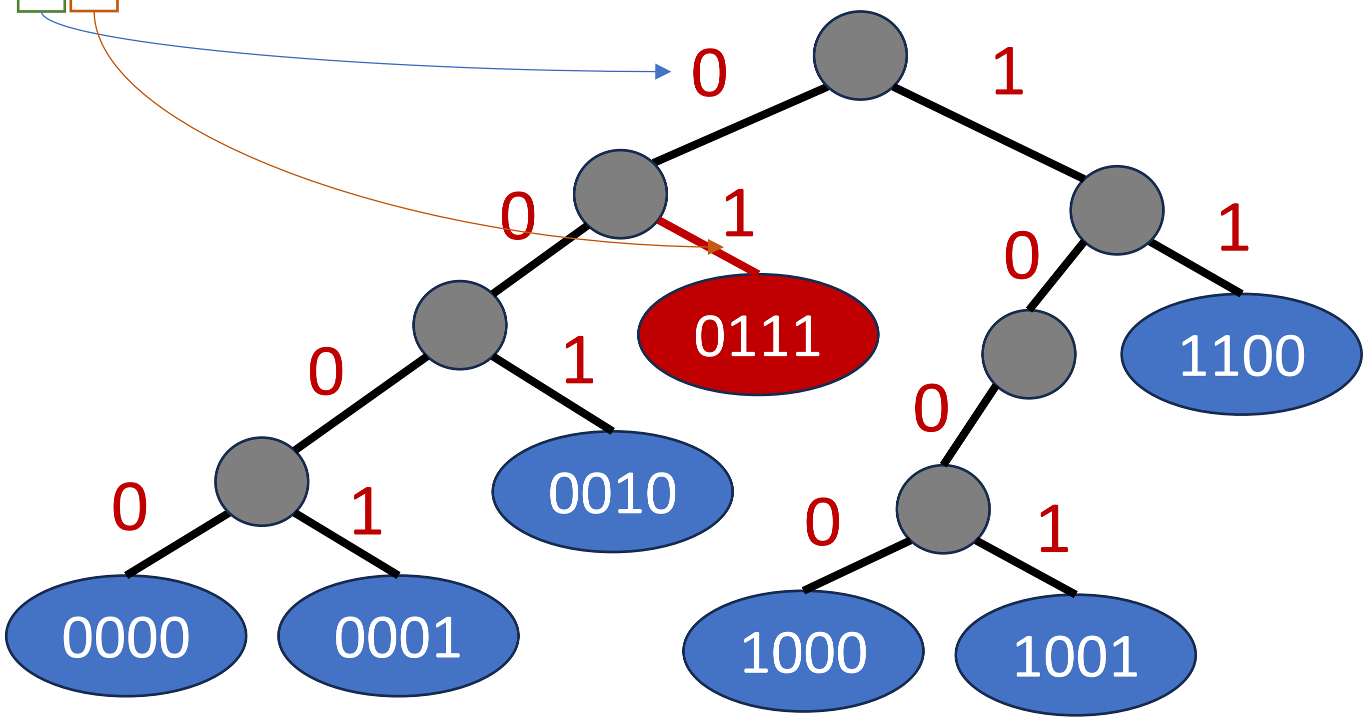
Note: The bits on edges are not stored in the data structure.





Operation: Insert

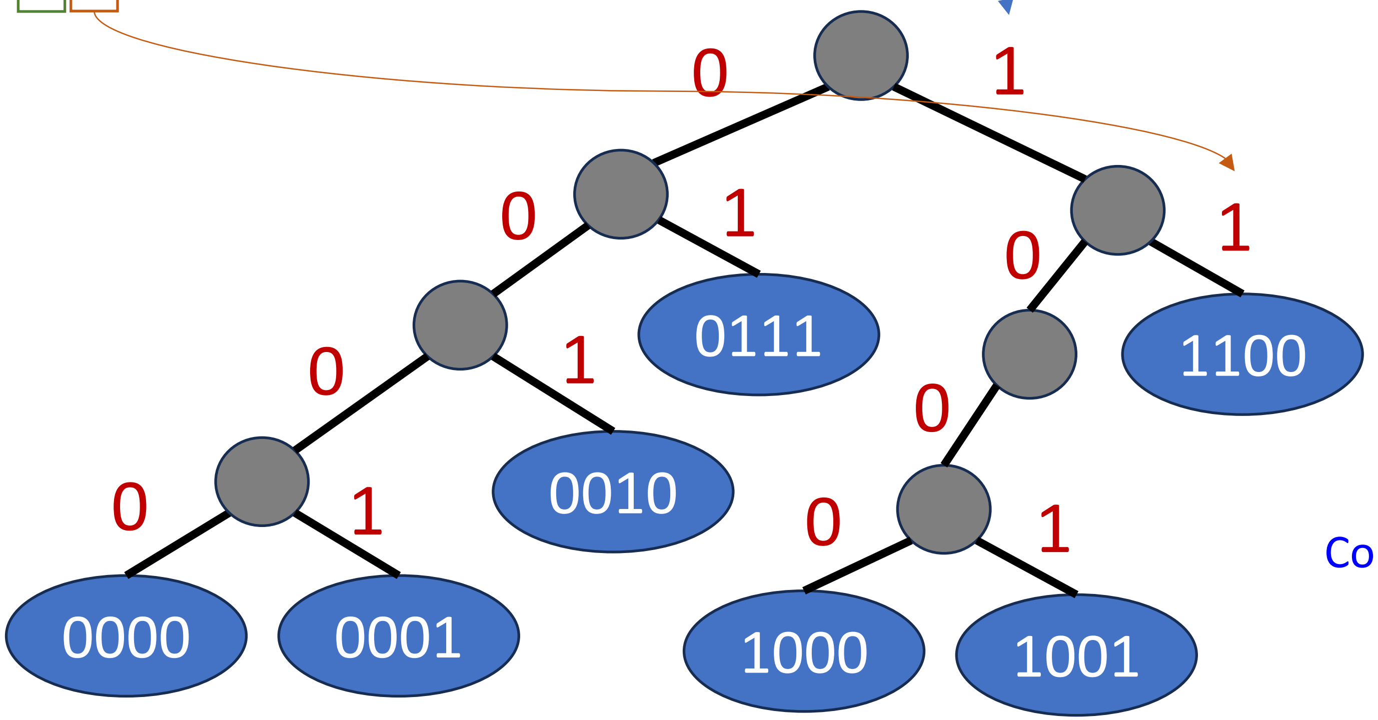
- Insert 0111.



Zero compare.

Operation: Insert

- Insert 1101.

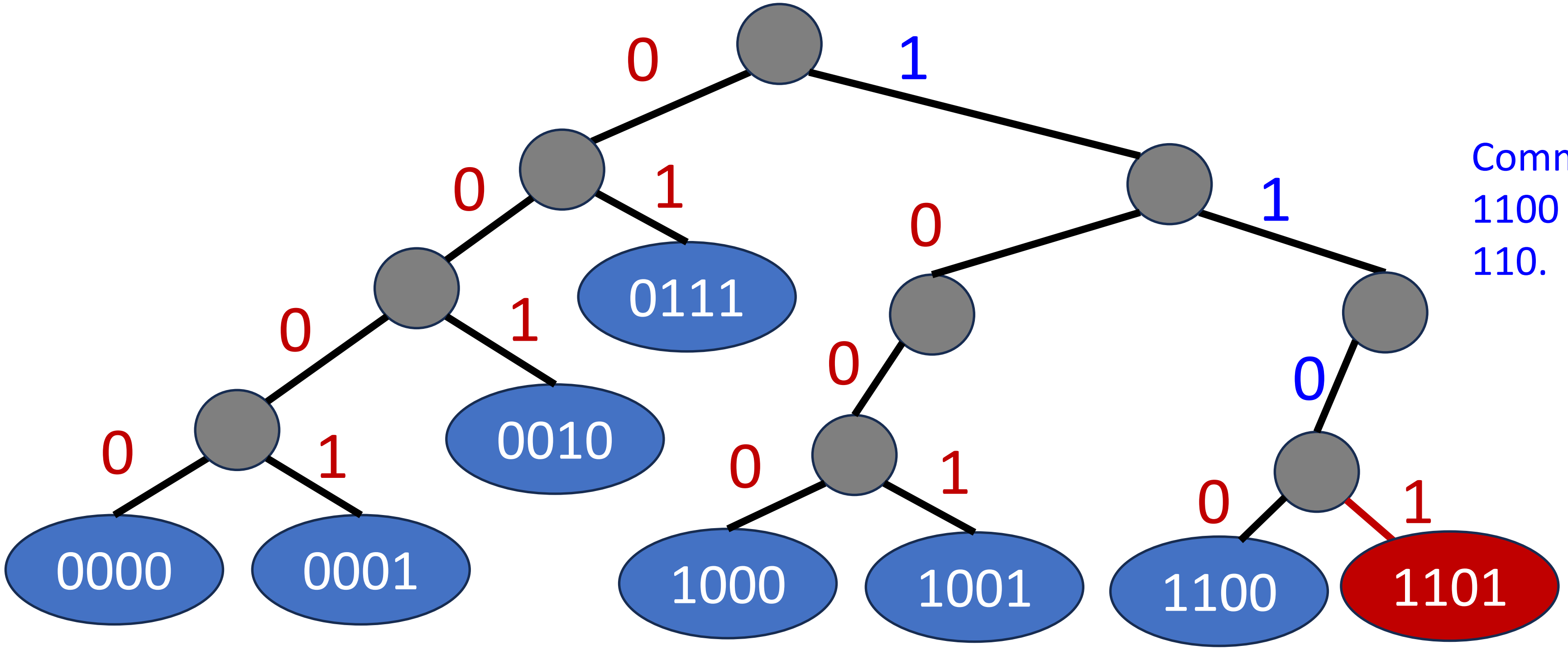


Compare.
Keys are different.

Common prefix of 1100 and 1101 is 110.

Operation: Insert

- Insert 1101.

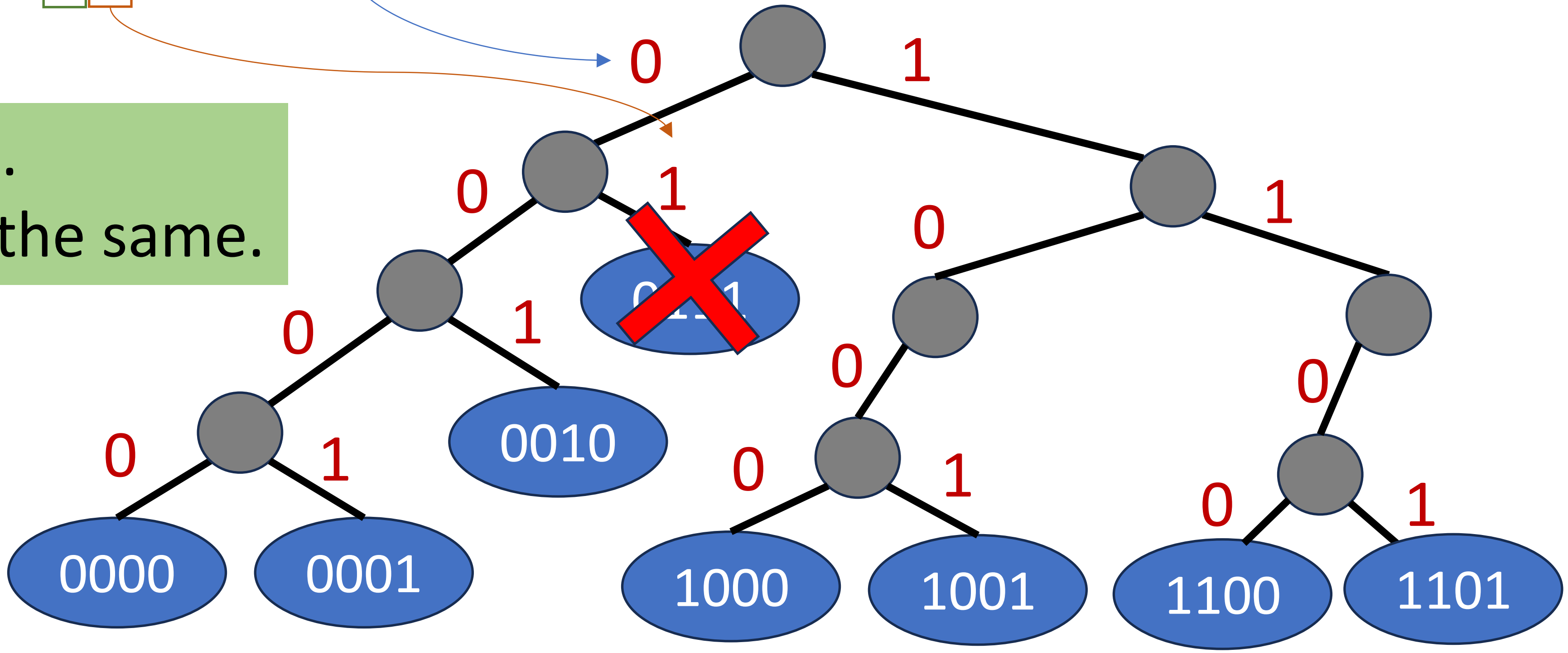


Common prefix of 1100 and 1101 is 110.

Operation: Delete

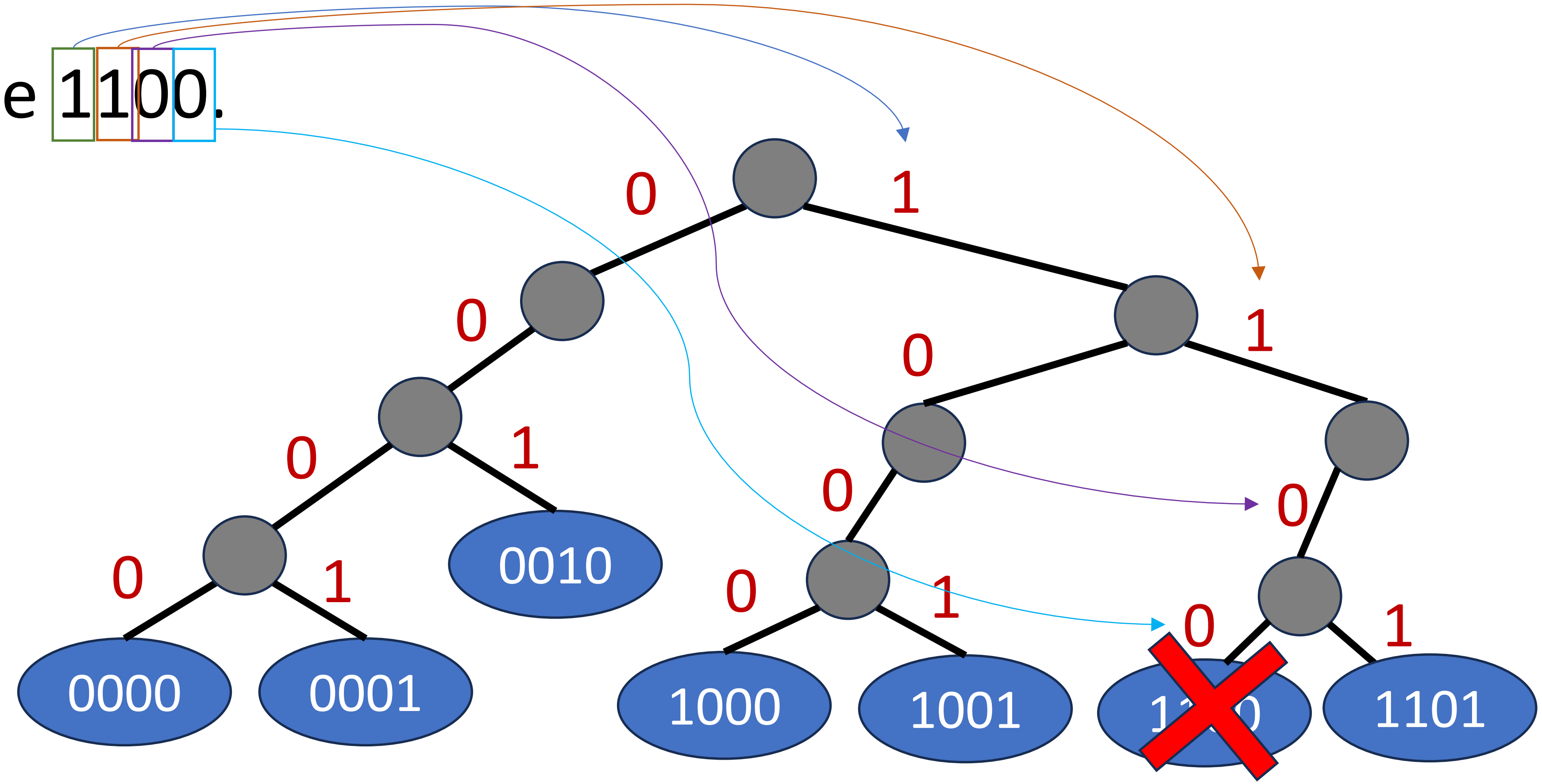
- Delete 0111.

Compare.
Keys are the same.



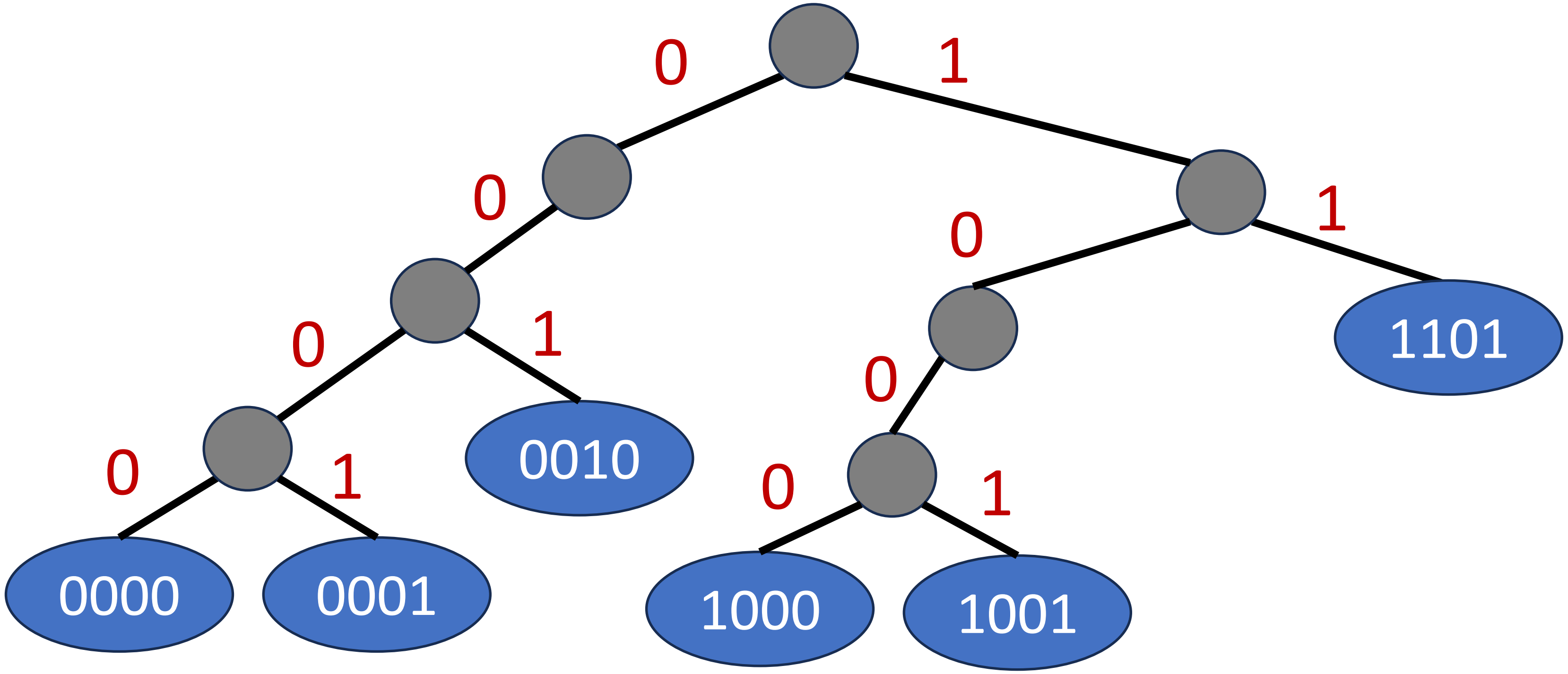
Operation: Delete

- Delete 1100.



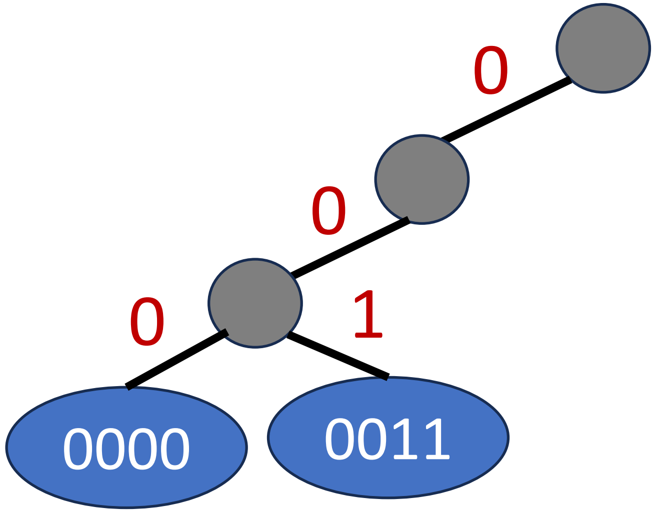
Operation: Delete

- Delete 1100.



Exercise

- Q7: Given the keys 0010, 1000, 1100, 1101. Following this order, please insert the keys into the following binary trie. What will be the keys in the element nodes of the 5th level of the resultant tree?



- Q8: (Continue Q7) After deleting the node whose key is 0011, what will be the keys in the element nodes of the 5th level of the resultant tree?

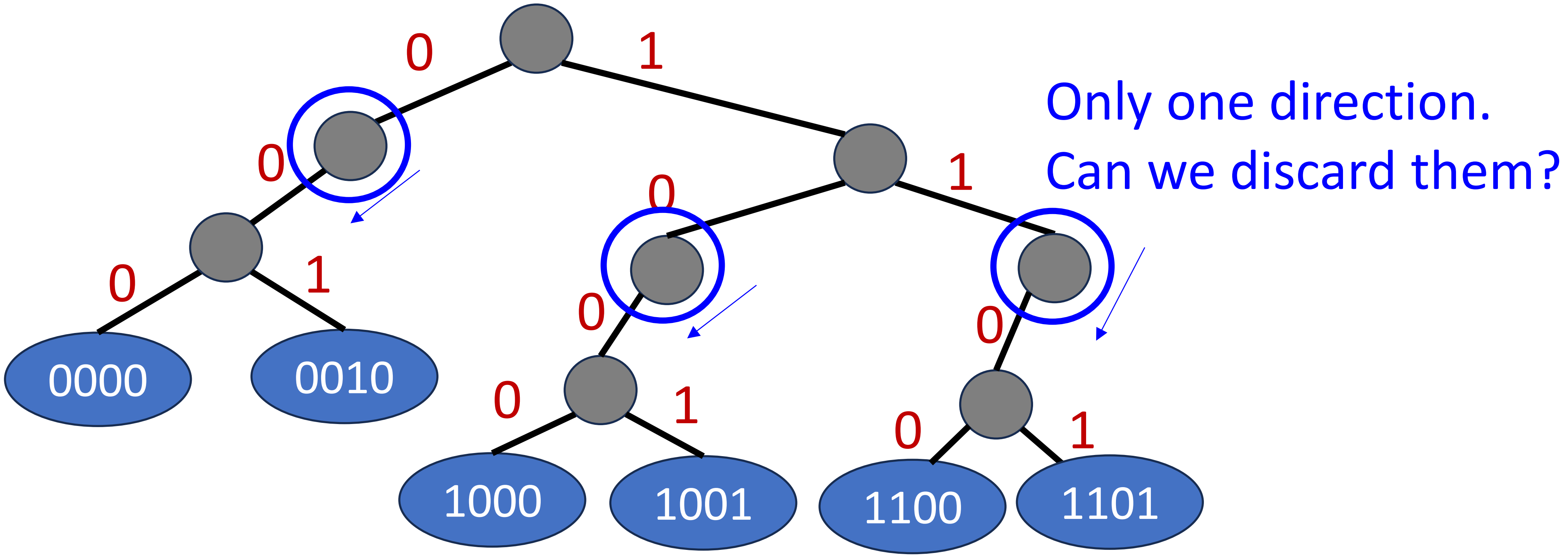
Please reply your answers of Q7-Q8 via the following link:

https://forms.gle/NyCVWyzZ6eXwVdYg9

Group members: 2~4 people

Discussion

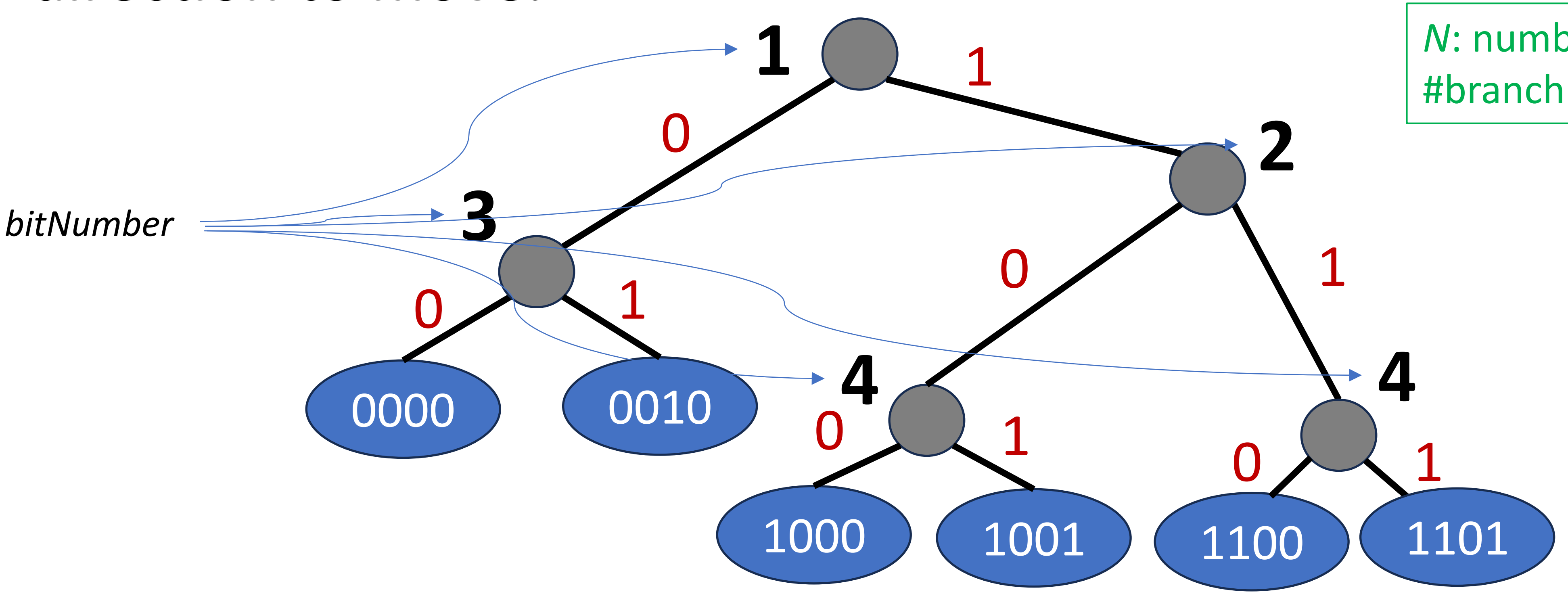
- 6 element nodes.
- 8 branch nodes.
- Too much overhead.



3. Compressed Binary Trie

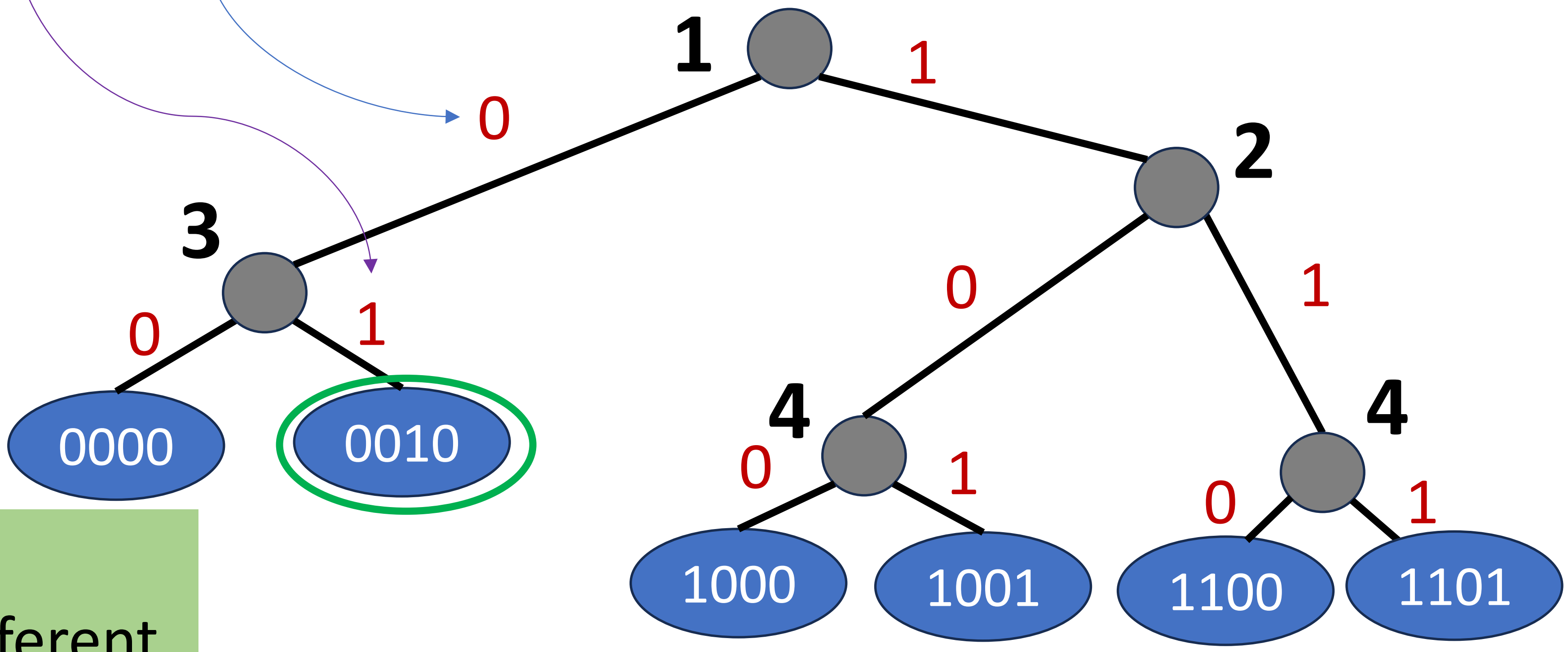
- No branch node whose degree is 1.
- Add a *bitNumber* field to each branch node
- *bitNumber* tells you which bit of the key to use to decide the direction to move.

N: number of element node
#branch nodes = $N - 1$



Operation: Insert

- Insert 0011.



Compare.
Keys are different.

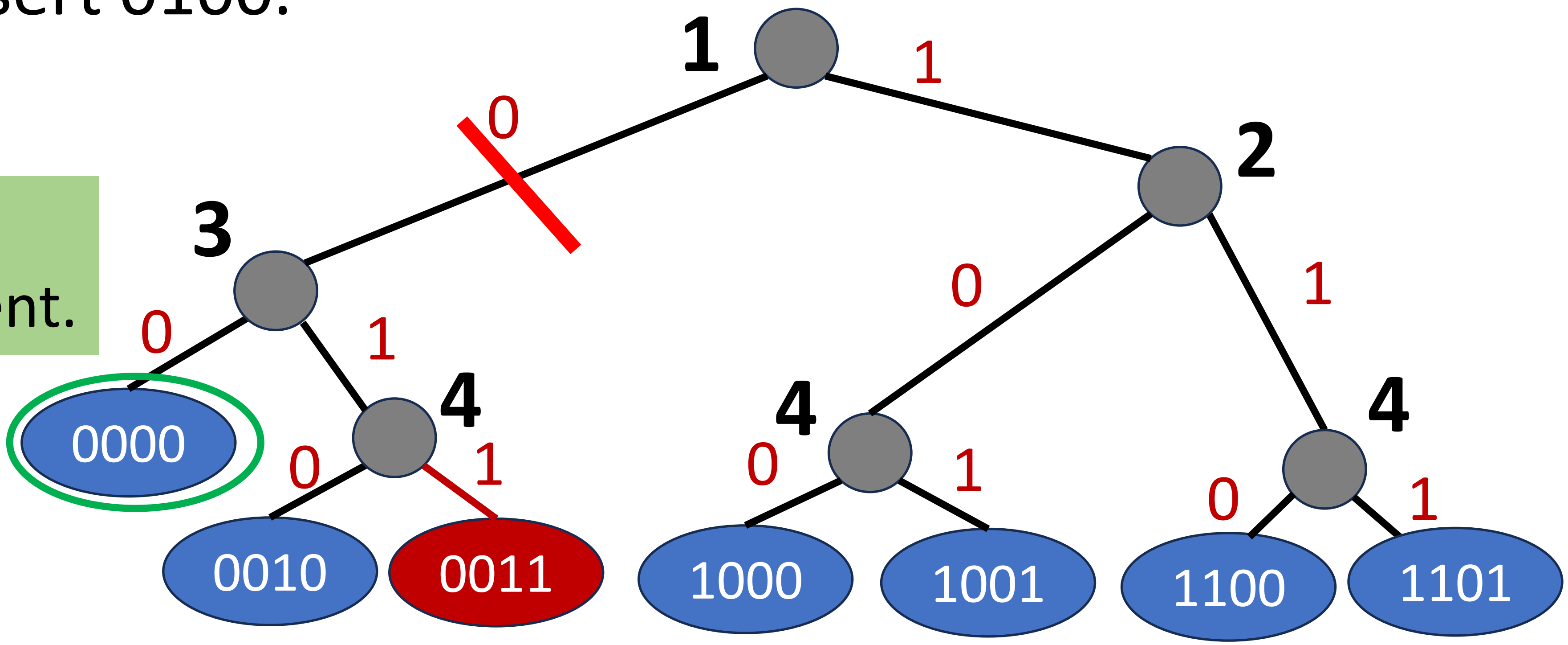
Common prefix of 0011 and 0010 is 001.

Operation: Insert

- Insert 0011.
- Then, insert 0100.

Compare.
Keys are different.

Common prefix of 0100 and 0000 is 0.
The 2nd bits of them are different.



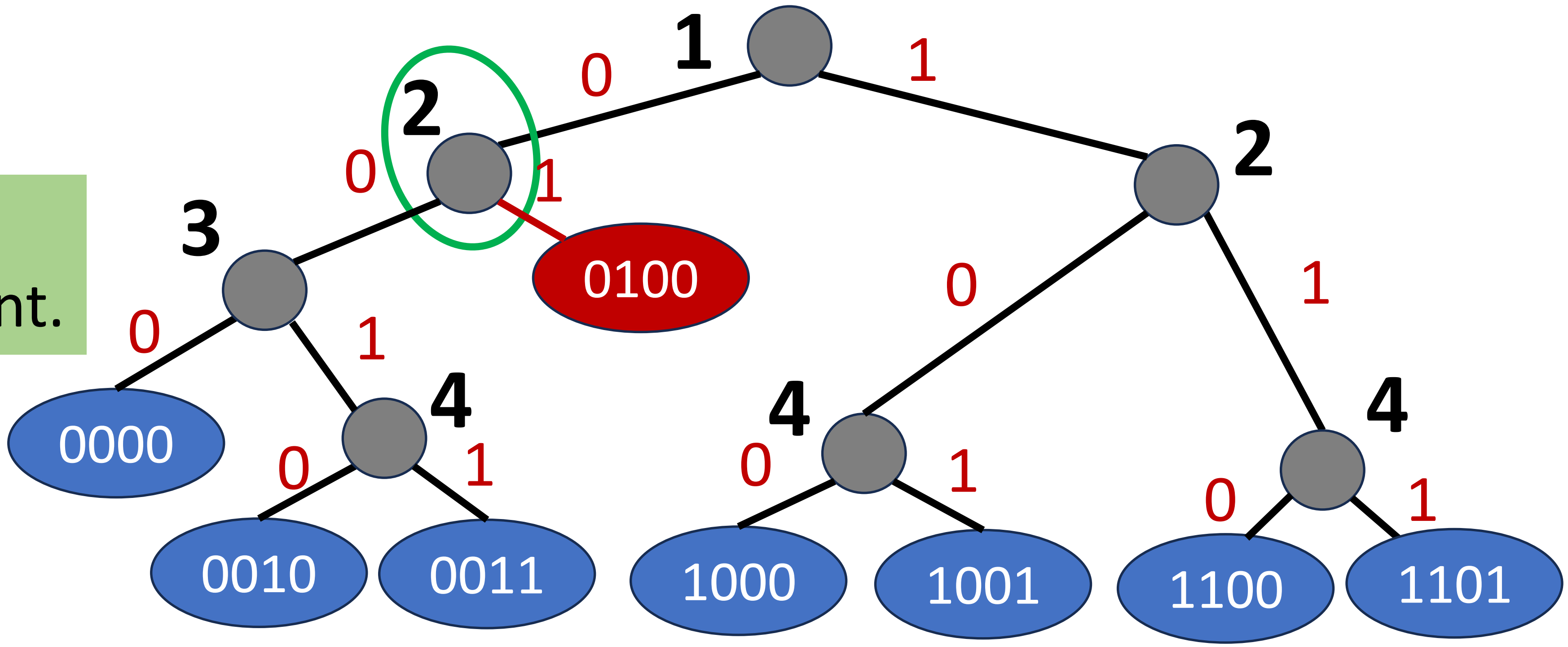
Common prefix of 0011 and 0010 is 001.
The 4th bits of 0011 and 0010 are different.

Operation: Insert

- Insert 0100.

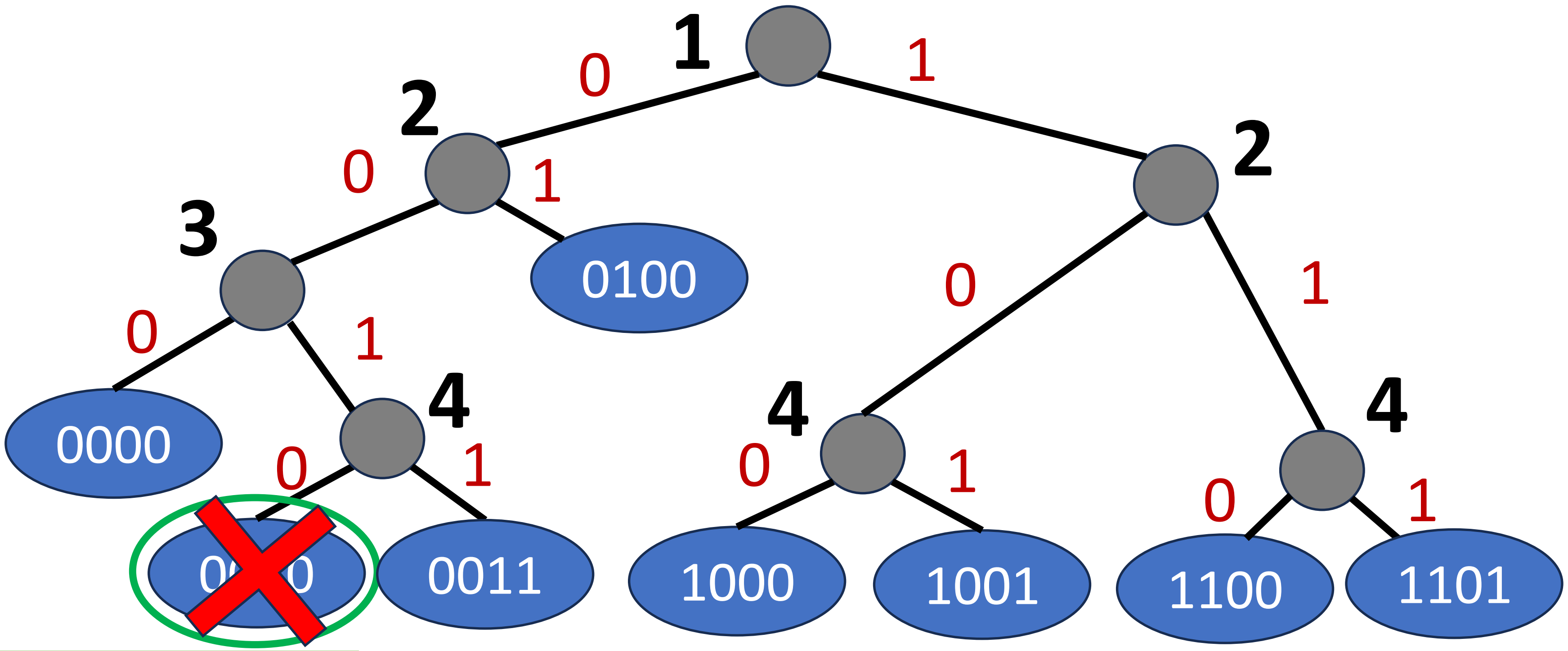
Compare.
Keys are different.

Common prefix of 0100
and 0000 is 0.
The <u>2nd bits</u> of them
are different.



Operation: Deletion

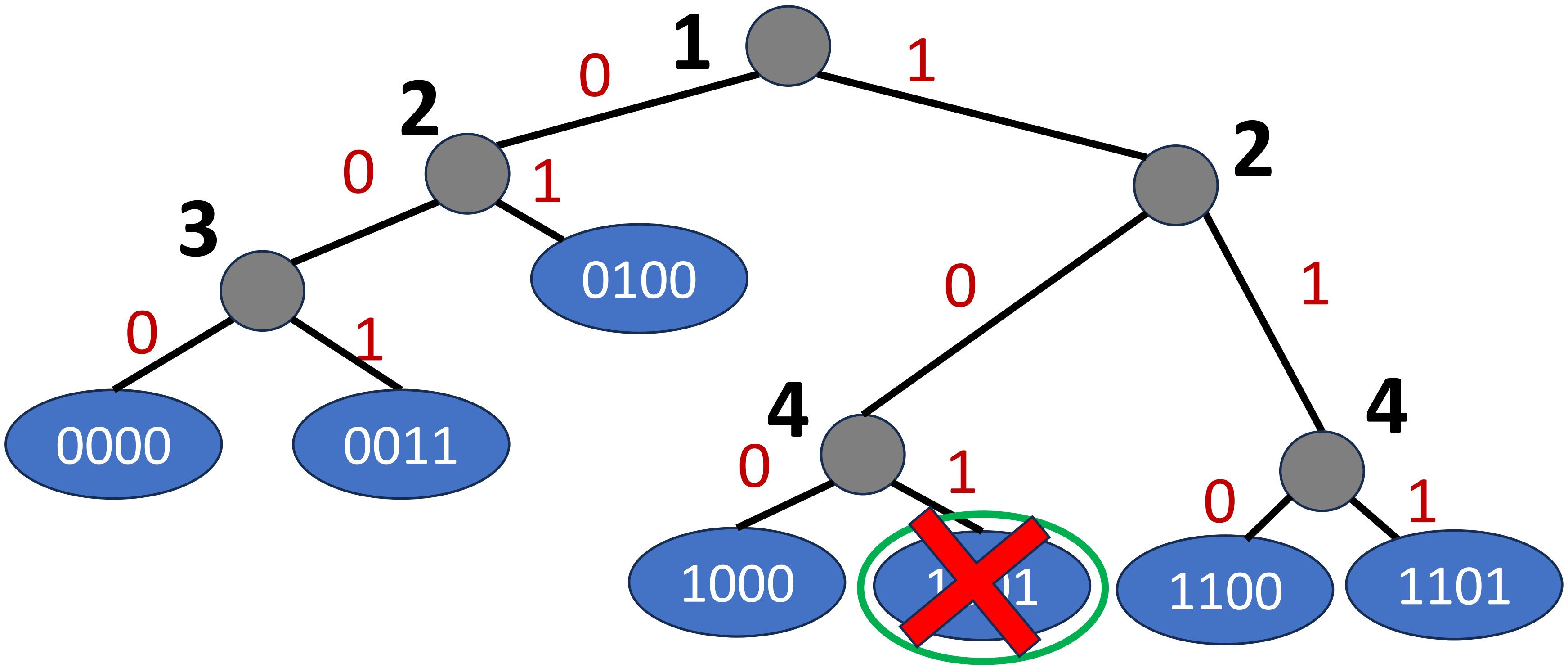
- Delete 0010.



Compare.
Keys are the same.

Operation: Deletion

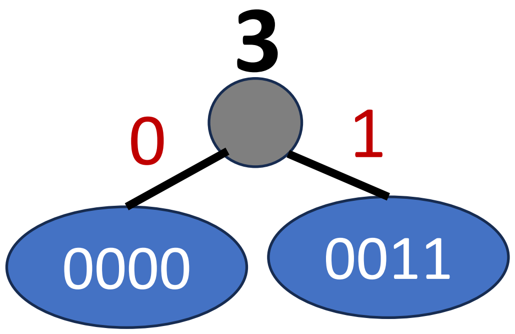
- Delete 1001.



Compare.
Keys are the same.

Exercise

- Q9: Given the keys 0010, 1000, 1100, 1101. Following this order, please insert the keys into the following compressed binary trie. What will be the keys in the element nodes of the 3rd level of the resultant tree?



Please reply your answers of Q9 via the following link:

https://forms.gle/NyCVWyzZ6eXwVdYg9

Group members: 2~4 people

Exercise

- Q10: (Continue Q9) After deleting the node whose key is 0011, what will be the keys in the element nodes of the 3rd level of the resultant tree?

Please reply your answers of Q10 via the following link:

https://forms.gle/NyCVWyzZ6eXwVdYg9

Group members: 2~4 people

Performance

- **Digital Search Tree** ✓

Reduce number of comparisons (decrease time)

- **Trie** ✓

Reduce required space for index (branch node)

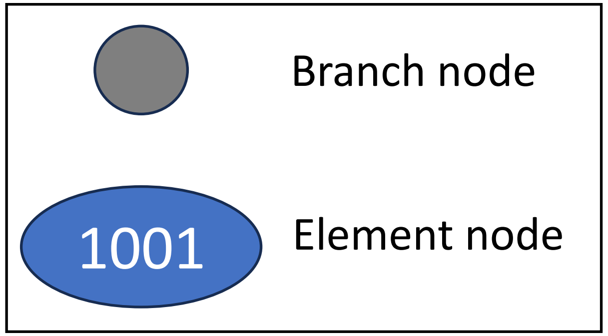
- **Compressed trie** ✓

Reduce required space for data (element nodes)

- **Patricia**

Performance is improved.

- Space ↓
- Time ↓



Summary

- Digital search tree
- Trie
- Compressed trie

- Operations:
  - Search
  - Insertion
  - Deletion