Apriori: Levelwise Search

1: \{a, d, e\}
2: \{b, c, d\}
3: \{a, c, e\}
4: \{a, c, d, e\}
5: \{a, e\}
6: \{a, c, d\}
7: \{b, c\}
8: \{a, c, d, e\}
9: \{c, b, e\}
10: \{a, d, e\}

Example transaction database with 5 items and 10 transactions.

Minimum support: 30%, i.e., at least 3 transactions must contain the item set.

All one item sets are frequent → full second level is needed.
1: \{a, d, e\}  
2: \{b, c, d\}  
3: \{a, c, e\}  
4: \{a, c, d, e\}  
5: \{a, e\}  
6: \{a, c, d\}  
7: \{b, c\}  
8: \{a, c, d, e\}  
9: \{c, b, e\}  
10: \{a, d, e\}  

Determining the support of item sets: For each item set traverse the database and count the transactions that contain it (highly inefficient).

Better: Traverse the tree for each transaction and find the item sets it contains (efficient: can be implemented as a simple doubly recursive procedure).
Apriori: Levelwise Search

Minimum support: 30%, i.e., at least 3 transactions must contain the item set.

Infrequent item sets: \{a, b\}, \{b, d\}, \{b, e\}.

The subtrees starting at these item sets can be pruned.
Generate candidate item sets with 3 items (parents must be frequent).

Before counting, check whether the candidates contain an infrequent item set.

• An item set with $k$ items has $k$ subsets of size $k - 1$.
• The parent is only one of these subsets.
The item sets \{b, c, d\} and \{b, c, e\} can be pruned, because

- \{b, c, d\} contains the infrequent item set \{b, d\} and
- \{b, c, e\} contains the infrequent item set \{b, e\}. 
Only the remaining four item sets of size 3 are evaluated.
Apriori: Levelwise Search

1: \{a, d, e\}
2: \{b, c, d\}
3: \{a, c, e\}
4: \{a, c, d, e\}
5: \{a, e\}
6: \{a, c, d\}
7: \{b, c\}
8: \{a, c, d, e\}
9: \{c, b, e\}
10: \{a, d, e\}

Minimum support: 30\%, i.e., at least 3 transactions must contain the item set.

Infrequent item set: \{c, d, e\}. 
Generate candidate item sets with 4 items (parents must be frequent).

Before counting, check whether the candidates contain an infrequent item set.
The item set \{a, c, d, e\} can be pruned, because it contains the infrequent item set \{c, d, e\}.

Consequence: No candidate item sets with four items.

Fourth access to the transaction database is not necessary.