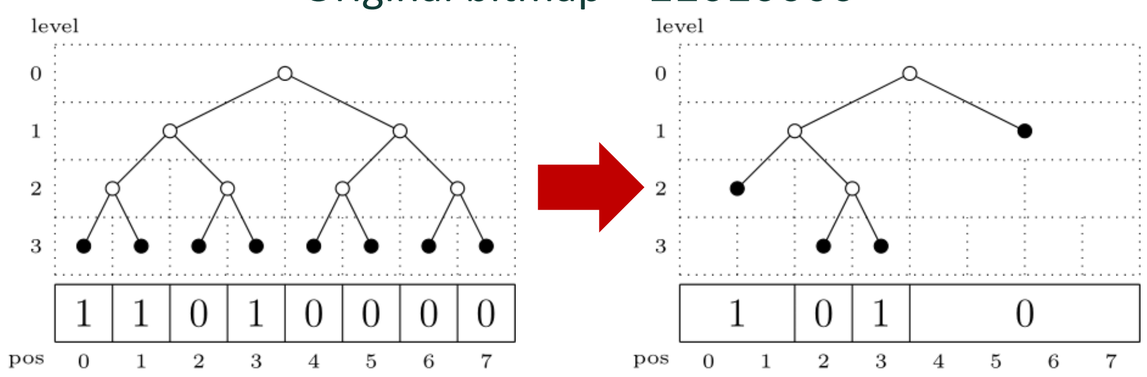
In-Place Updates in Tree-Encoded Bitmaps

Marcellus Prama Saputra, Eleni Tzirita Zacharatou, Serafeim Papadias, and Volker Markl

Tree-Encoded Bitmaps (TEBs)

Original bitmap = 11010000

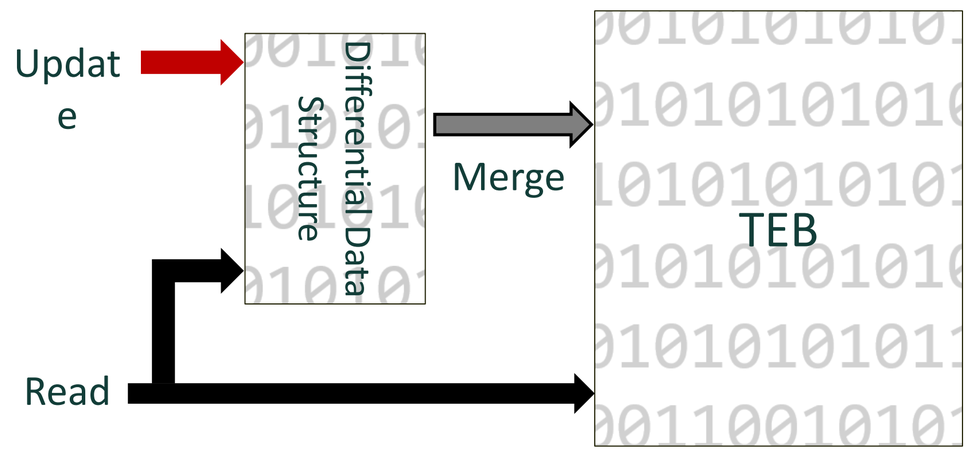


T = 1100100

L = 0101

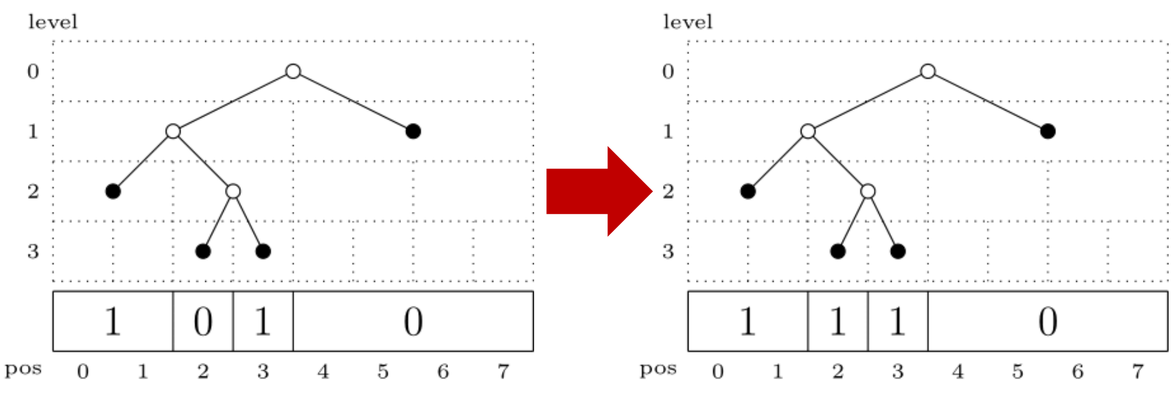
T represents the structure of the TEB.
L contains the labels of every leaf node.

Differential Updates



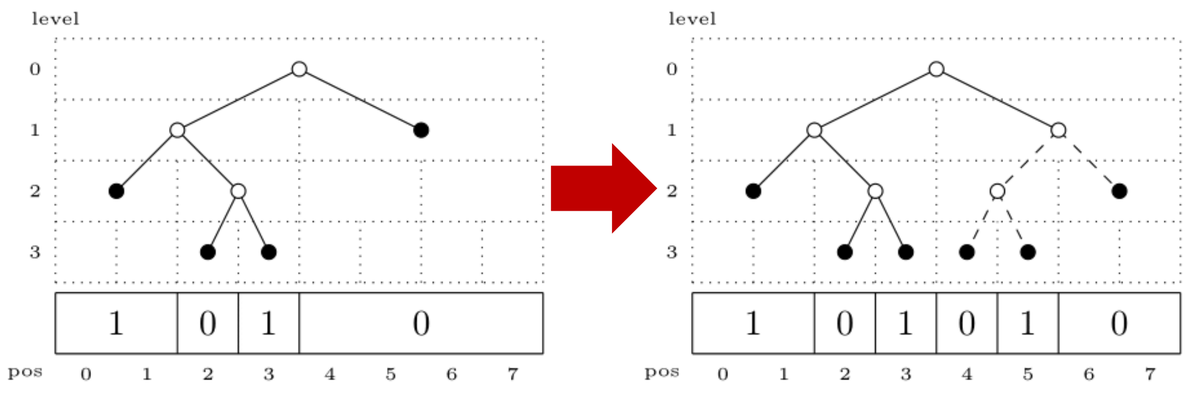
The current update approach uses an auxiliary data structure, i.e., a *differential data structure*, to store updates.
The differential data structure is merged with the TEB to apply updates.
Additional memory and read overhead.

Run-Forming Updates



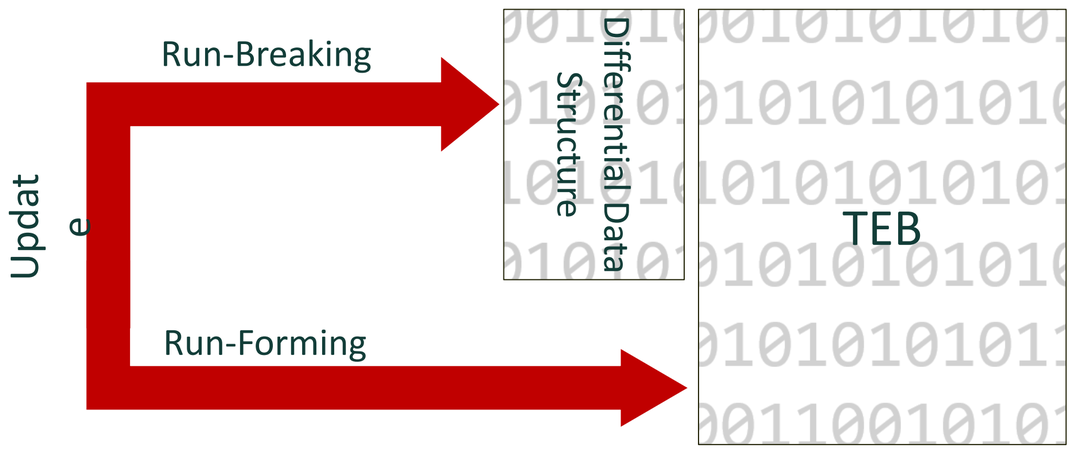
Update affects a leaf node that represents an individual bit.
Perform update by changing the label of the leaf node.
Only L needs to be modified.

Run-Breaking Updates



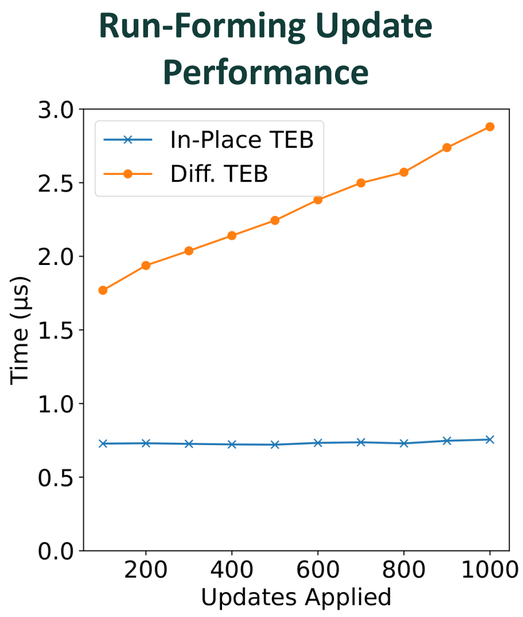
Update affects a leaf node that represents a run.
Perform update by replacing leaf node with a subtree.
Both T and L need to be modified.

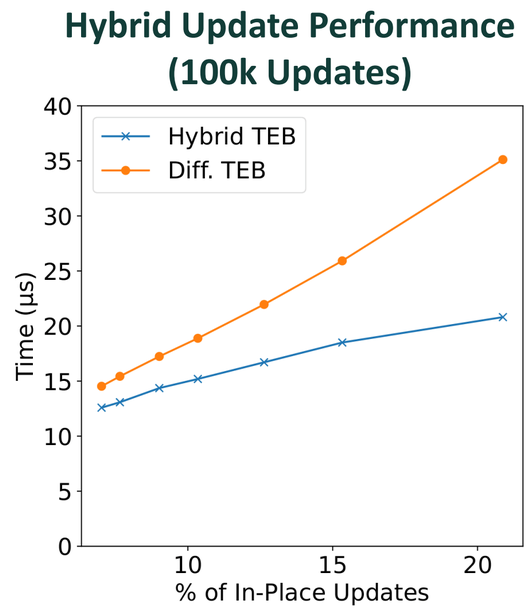
Hybrid Updates



Run-forming updates are fast, while run-breaking updates are slow.
The hybrid approach achieves the best of both worlds.
Perform run-forming updates in-place, and store run-breaking updates in a differential data structure.
Smaller differential data structure as fewer updates are stored.

Experimental Results





Data: Randomly generated bitmaps (1 million bits long) and updates.

Compared to differential updates:

- Run-forming updates are ≈ 3X faster.
- Hybrid updates are at least as fast.
- More run-forming updates => wider performance difference.
- With hybrid updates, the total TEB size is 4-9% smaller.

Technische Universität Berlin

IT-UNIVERSITETET I KØBENHAVN

DASYA