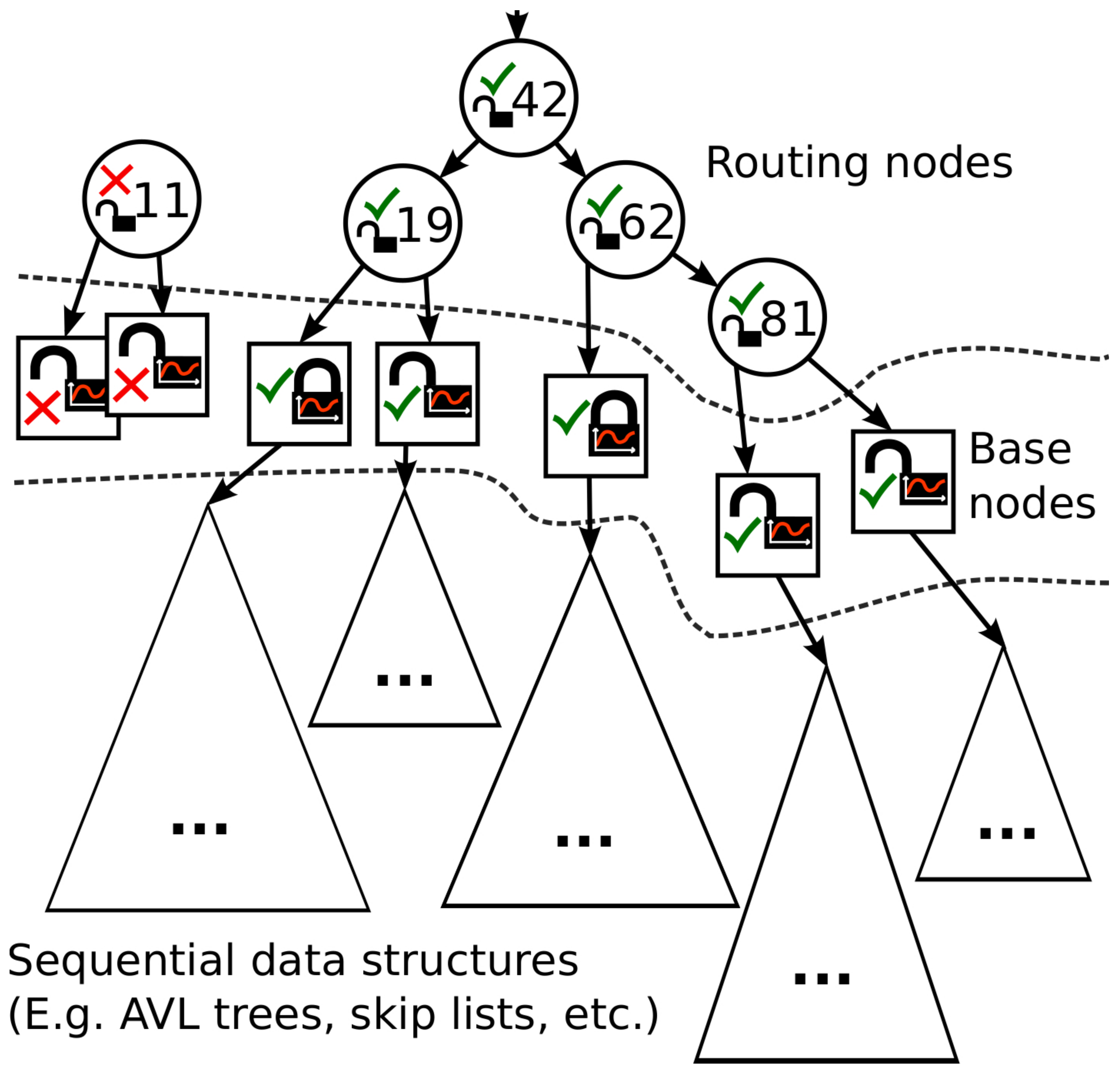
UPPSALA UNIVERSITET

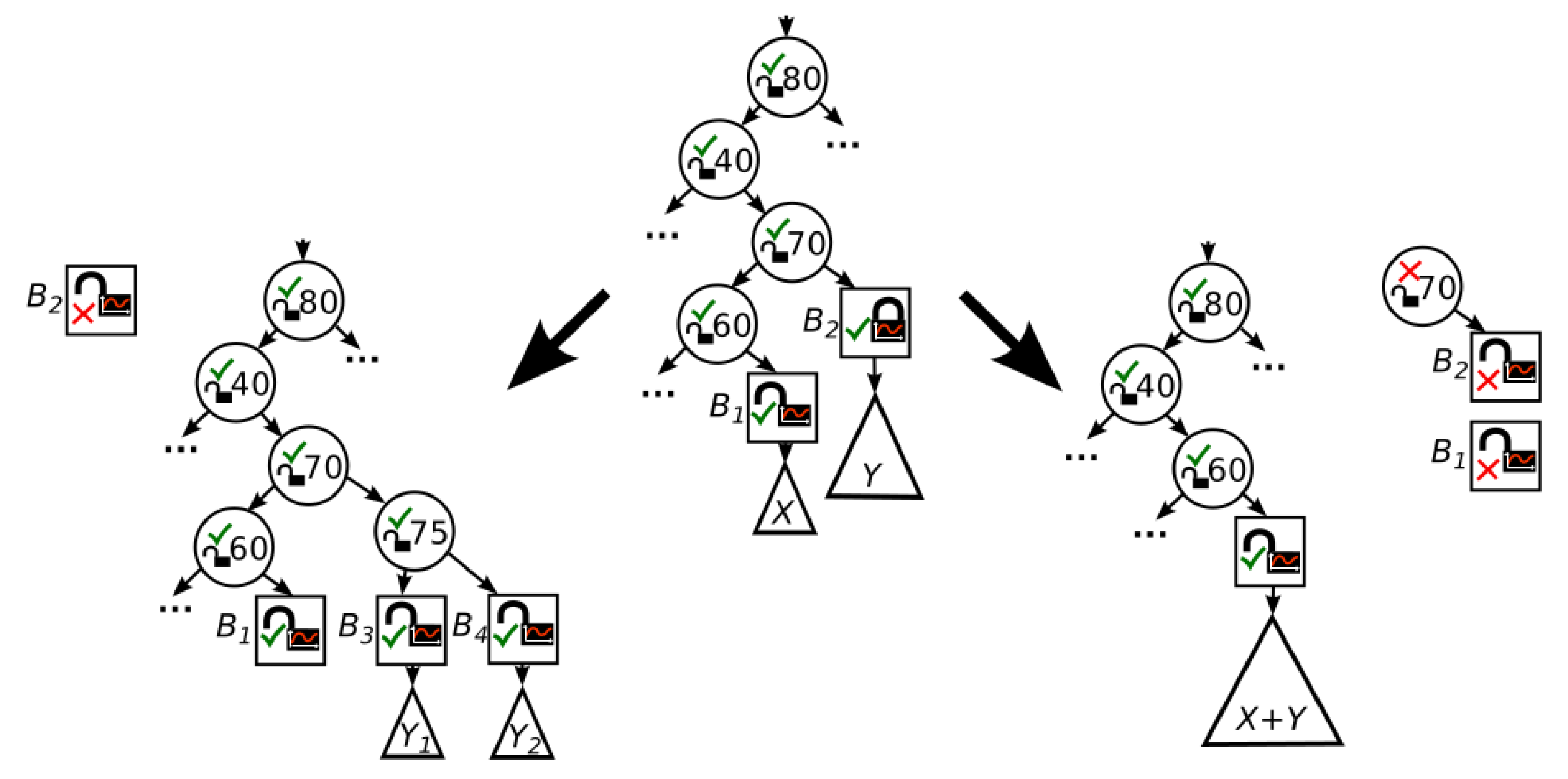
Contention Adapting Search Trees (CA trees)

A concurrent data structure for sets and maps


42
11
19
62
81
Routing nodes
Base nodes
...
Sequential data structures
(E.g. AVL trees, skip lists, etc.)

How does it work?

- Information about contention and operations that benefit from fewer base nodes is collected in the base node locks (see right)
- Splits a base node when the estimated contention is above a threshold and joins two base nodes when the estimated contention is below a threshold (see down)



(a) After a split (b) Initial (c) After a join

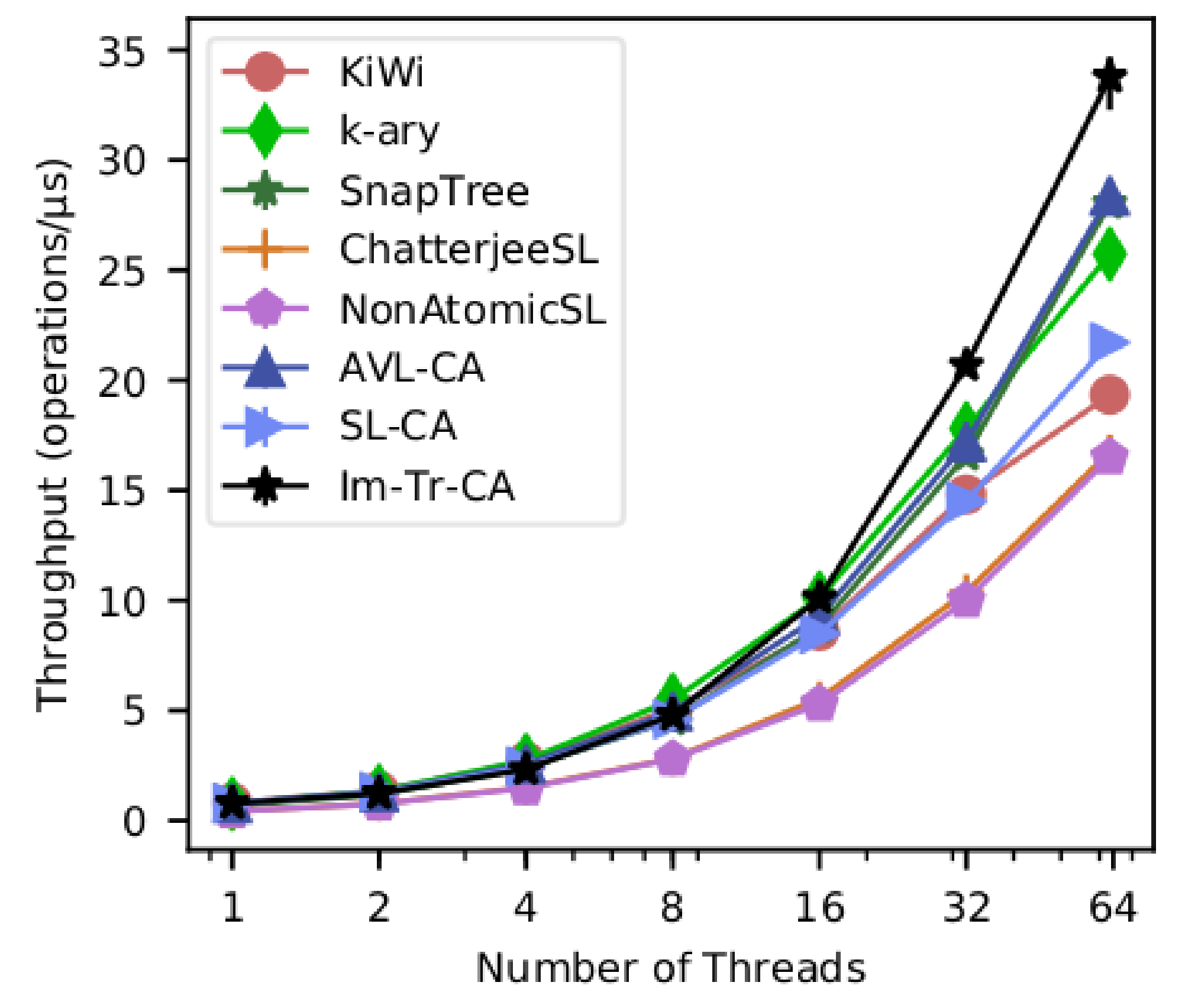
Why use CA trees?

- **Efficient and scalable** insert, remove, bulk updates, linearizable range queries etc
- **Low sequential overhead**
- **Flexible** since skip lists, AVL trees, etc. can be plugged in as sequential data structures
- **Self-adapting to fit access pattern**
- **Very short traversal of mutable data** when mutable pointer to immutable data structure is used as the sequential data structure
  - Gives **excellent performance for linearizable range queries** (even when they are very large)

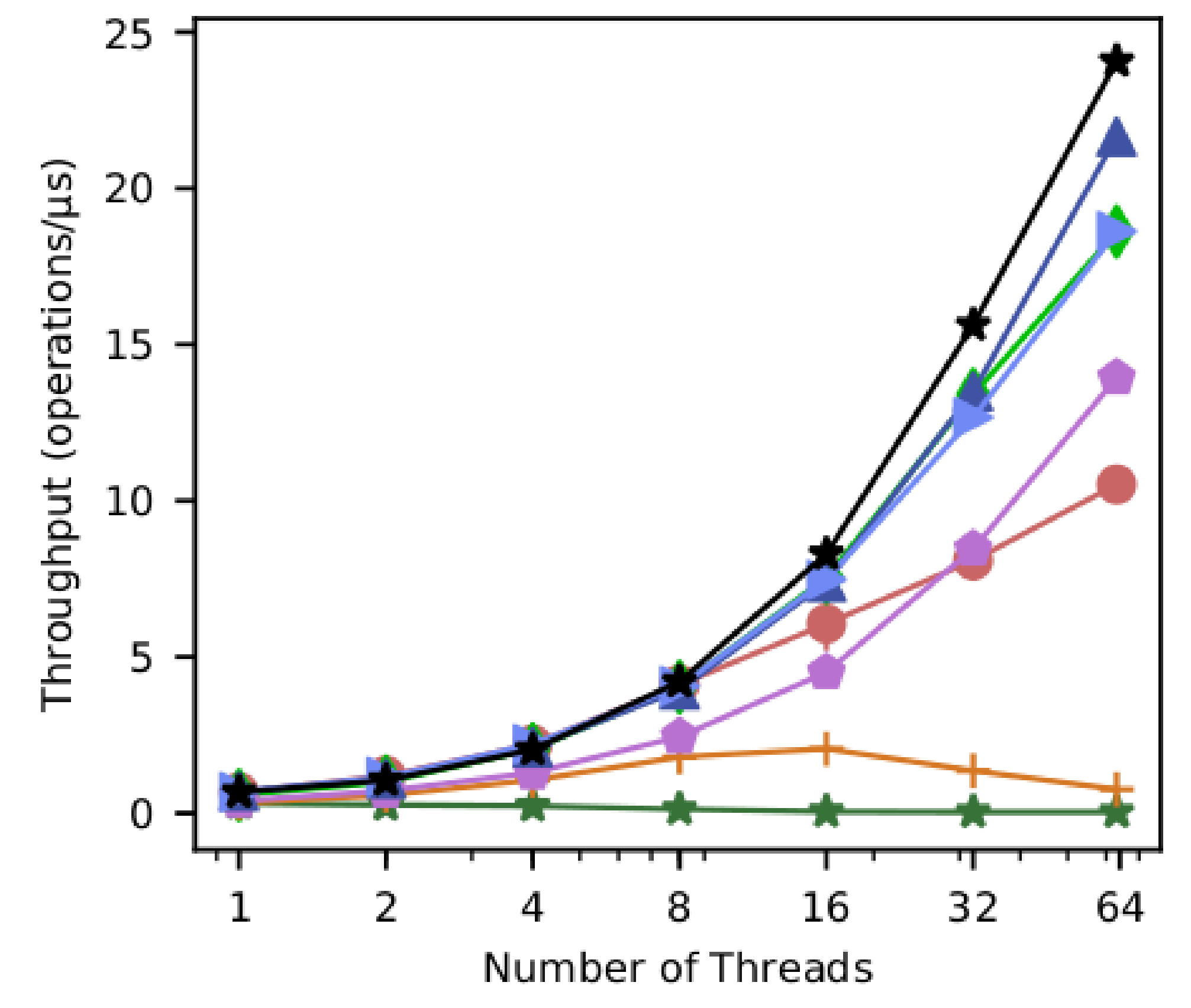
Experiments

w:x% r:y% q:z%-r where x% insert and remove operations, y% lookup operations and x% range queries with maximum range size r. Number of items in set ≈ 500000.

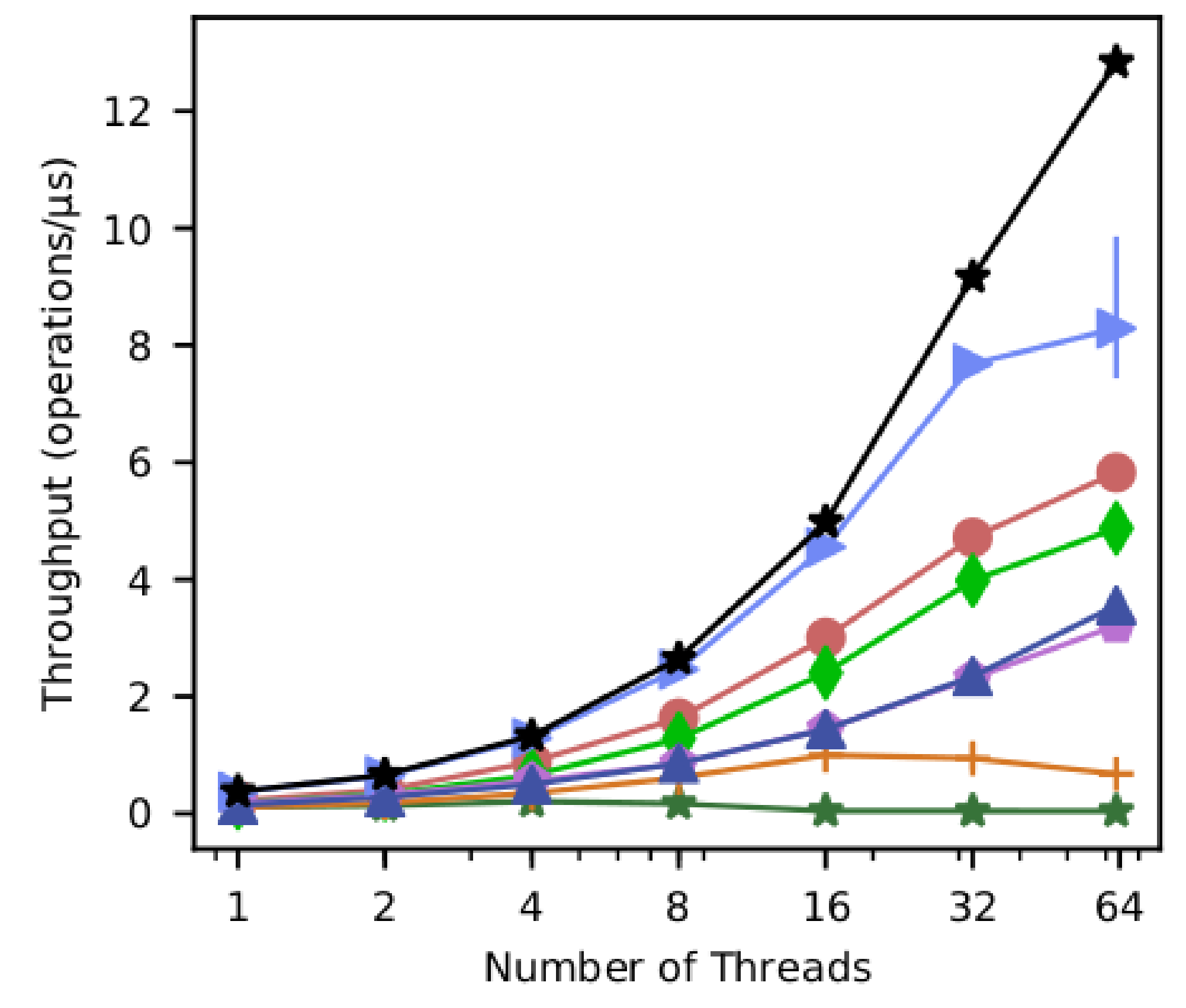
KiWi = From “KiWi: A Key-Value Map for Scalable Real-Time Analytics, PPoPP’2016”
k-ary = From “Range queries in non-blocking k-ary search trees, OPODIS’2012”
SnapTree = From “A practical concurrent binary search tree, PPoPP’2010”
ChatterjeeSL = From “Lock-free linearizable 1-dimensional range queries, ICDCN’2017”
NonAtomicSkipList = ConcurrentSkipListMap rom Java’s standard library
AVL-CA = CA tree using AVL tree as sequential data structure, LCPC’2015
SL-CA = CA tree using skip list with fat nodes as sequential data structure, LCPC’2015
Im-Tr-CA = CA tree using a mutable reference to immutable treap as sequential data structure (ICCSW’2017)



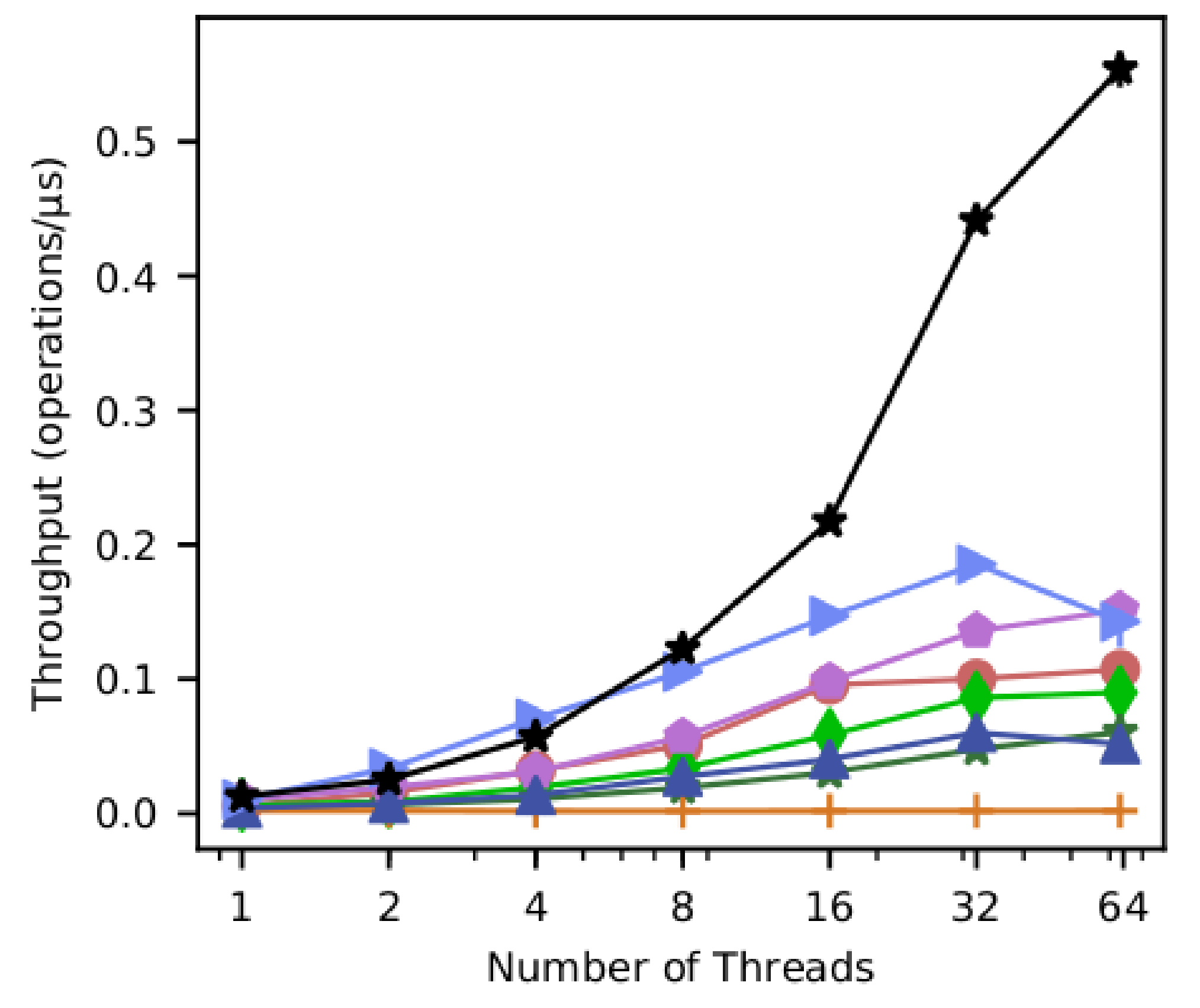
(a) w:20% r:80%



(b) w:20% r:55% q:25%-10



(c) w:20% r:55% q:25%-1000



(d) w:20% r:55% q:25%-100000

Publications

- Faster Concurrent Range Queries with Contention Adapting Search Trees Using Immutable Data, ICCSW’2017
- Efficient Support for Range Queries and Range Updates Using Contention Adapting Search Trees, LCPC’2015
- Contention Adapting Search Trees. ISPDC’2015
- More Scalable Ordered Set for ETS Using Adaptation, ACM Workshop on Erlang, 2014

More info
http://www.it.uu.se/research/group/languages/software/ca_tree

Kjell Winblad
kjell.winblad@it.uu.se
http://winsh.me