SBSA: A Constant-Time Story

Aaron Cattell

June 18, 2025

Chapter 1: The Wall We Keep Hitting

For decades, we've organized digital structures — tasks, files, events — using comparisons. Binary trees, heaps, skip lists, and B-trees dominate. But they all hit a wall: insertions slow down, scale breaks balance.

Chapter 2: A Simple But Different Idea

What if we didn't compare at all? What if each item had a known address? That's SBSA:

$$(slot,\ thickness,\ width)$$

Slot = priority, Thickness = queue, Width = float key (e.g. time). No search. Just logic.

Chapter 3: From Concept to Code

C++ and Python implementations followed. Files are stored at: `storage/slot_2/layer_3/file_5.5.txt` Then it became:

- A task scheduler
- A quantum circuit queue (Qiskit)
- A benchmark vs heap

Chapter 4: What SBSA Replaces

SBSA can replace:

- Priority queues
- B-trees, skip lists
- File indexers and schedulers

Chapter 5: Benchmarks — Flat Is Fast

10,000 task inserts:

System	Time (sec)
Heap	0.080
SBSA	0.005

SBSA stays flat: $O(1)$ performance.

Chapter 6: Real-World Use Cases

- Log file partitioning
- Task scheduling
- Event streaming
- Hybrid quantum/classical execution

Chapter 7: Quantum Job Queue

With Qiskit:

```
sbsa.write("High", 0, 3.5, circuit)
→ storage/slot_2/layer_0/file_3.5.qasm
```

Used for scheduling and executing circuits logically.

Chapter 8: The SBSA Mindset

¿ Don't search. Know the spot before you start.
It's deterministic. Spatial. Constant-time.

Chapter 9: Community and Traction

After 4 days:

- 700+ views
- 75+ clones
- 60+ active dev testers

SBSA is drawing early adopters and researchers.

Chapter 10: Where It's Going

Next:

- JSON/binary formats
- Compaction and pruning
- Rust/Go ports
- Full CLI and slot browsers

Final Words

SBSA began with a question: "Why log(n)?"
The answer became a model. A benchmark. A tool.

— *Aaron Cattell,*