Quantum Algorithms for Evaluating MIN-MAX Trees

Richard Cleve Dmitry Gavinsky D. L. Yonge-Mallo

Institute for Quantum Computing,
University of Waterloo

January 30, 2008
TQC – Tokyo, Japan

Motivation

- Why do we care about algorithms for MIN-MAX trees, anyway?

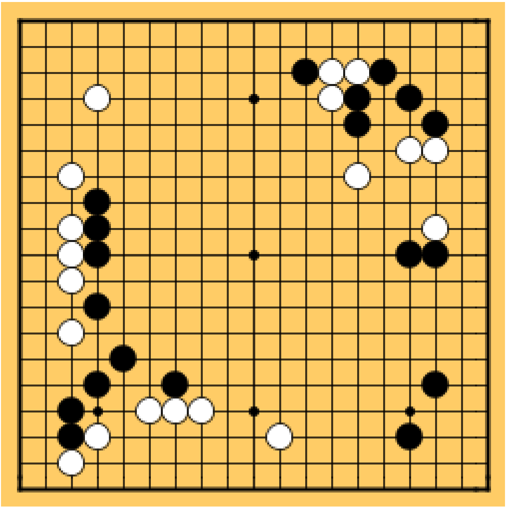
- What is so special about the quantum algorithms for MIN-MAX trees that I'm about to present?
  - The ideas behind them don't work in a classical setting!
  - Conversely, the classical ideas don't work in a quantum setting!

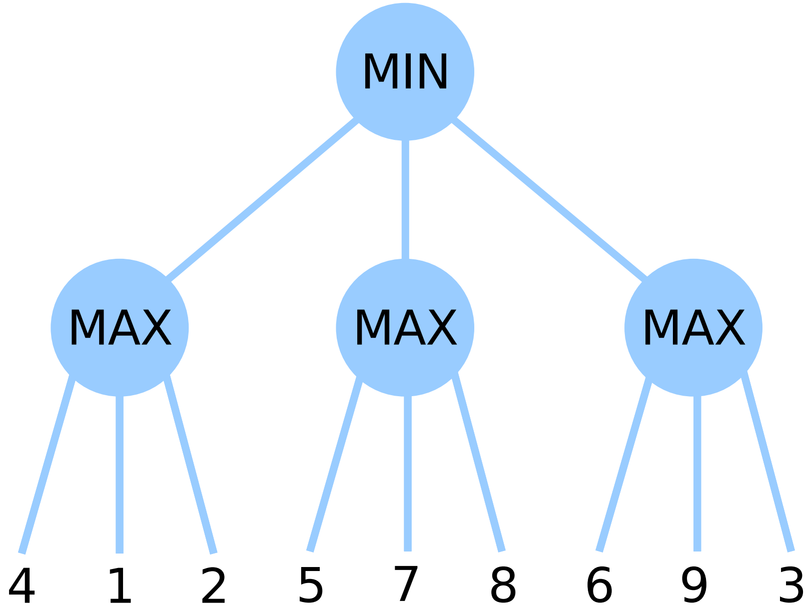
Why do we care about MIN-MAX trees?

MIN-MAX trees arise in the analysis of deterministic games of perfect information between two players who alternate taking turns

Why do we care about MIN-MAX trees?

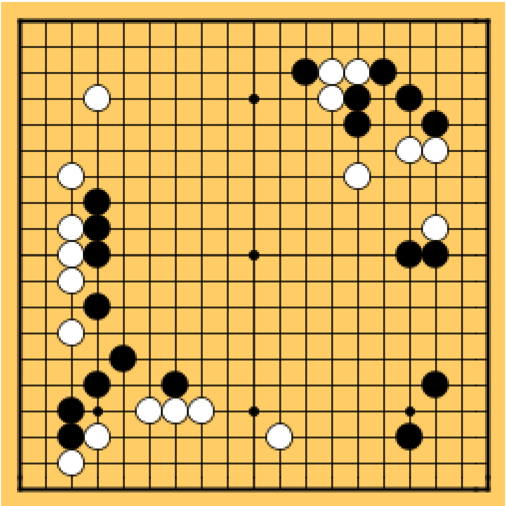
MIN-MAX trees arise in the analysis of deterministic games of perfect information between two players who alternate taking turns

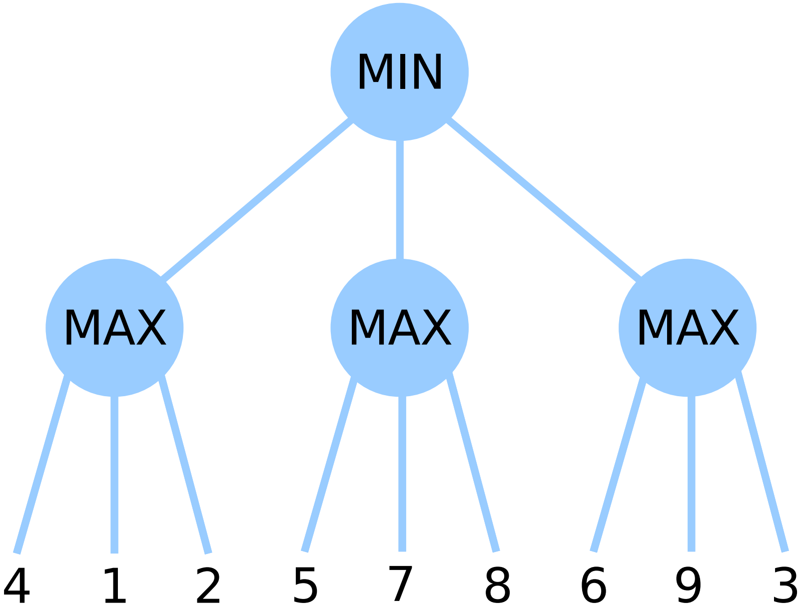




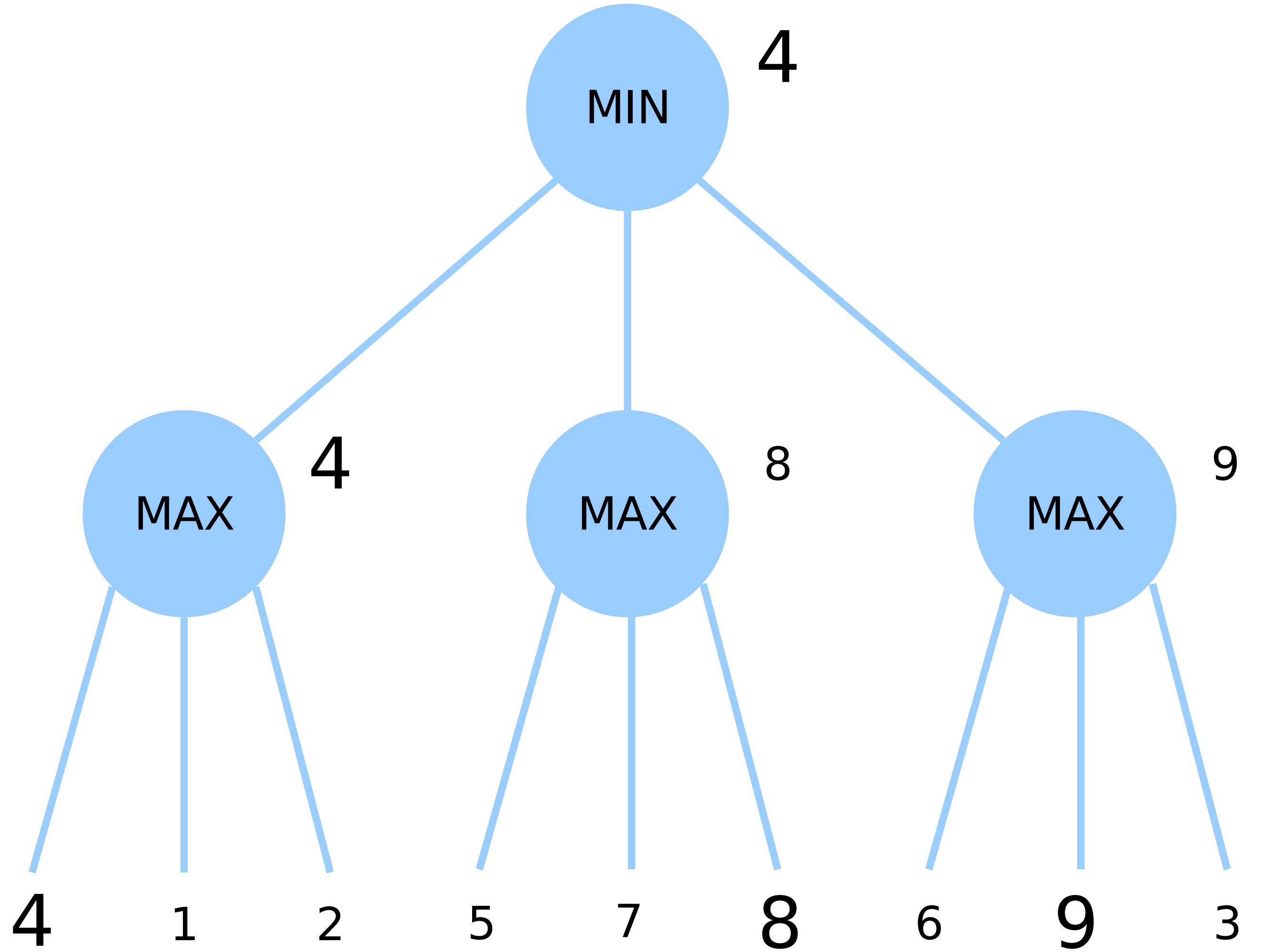
What is a MIN-MAX tree?

- internal nodes are MIN and MAX gates at alternating levels;
- leaves $x_1, \dots, x_N$ take on values from some ordered set;
- value is value of root as a function of $x_1, \dots, x_N$.

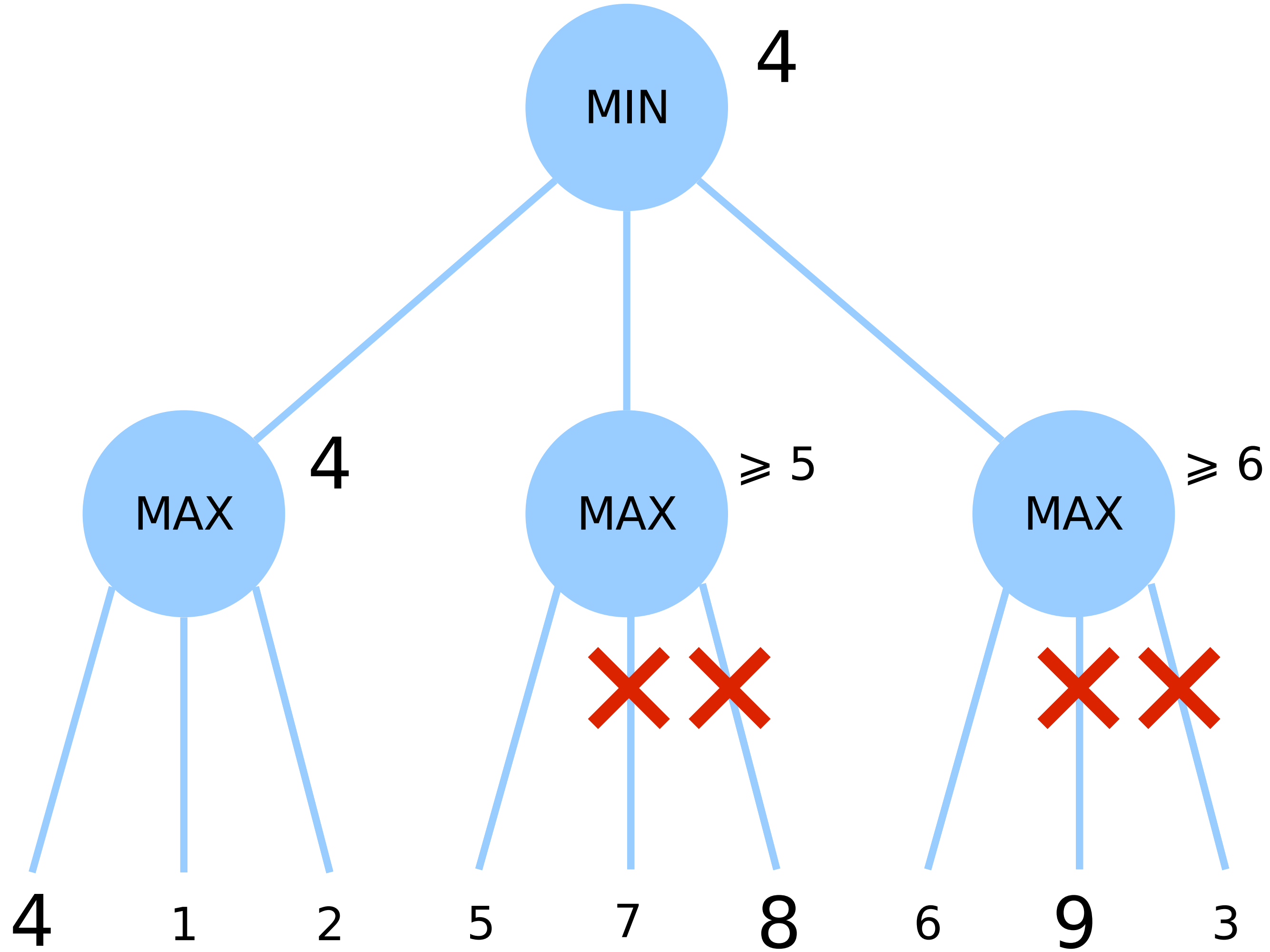




Evaluating a MIN-MAX tree

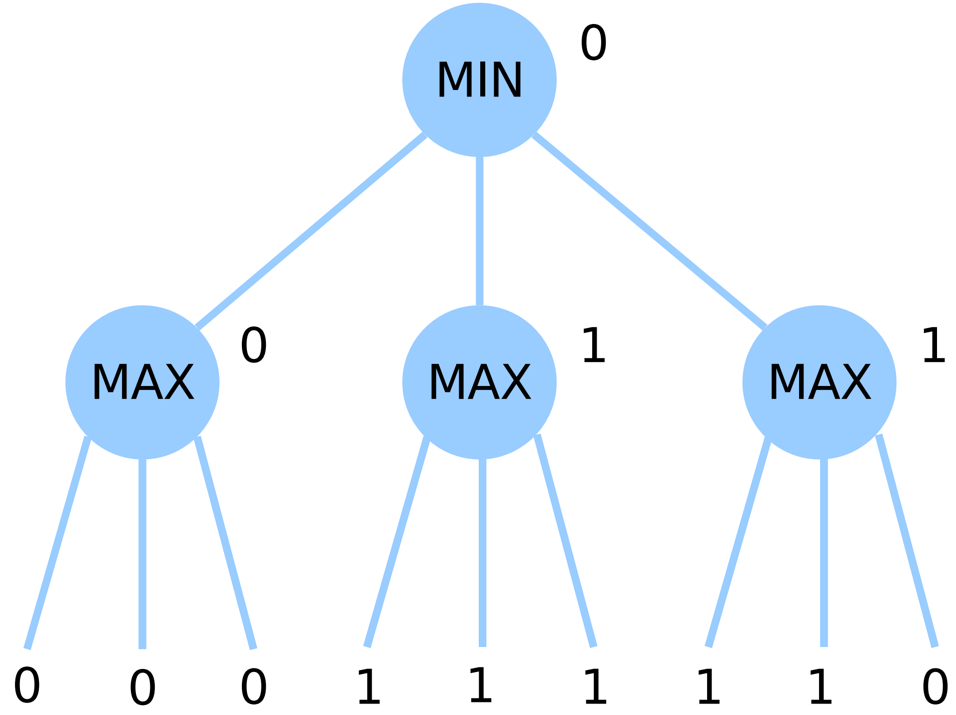


Alpha-beta pruning



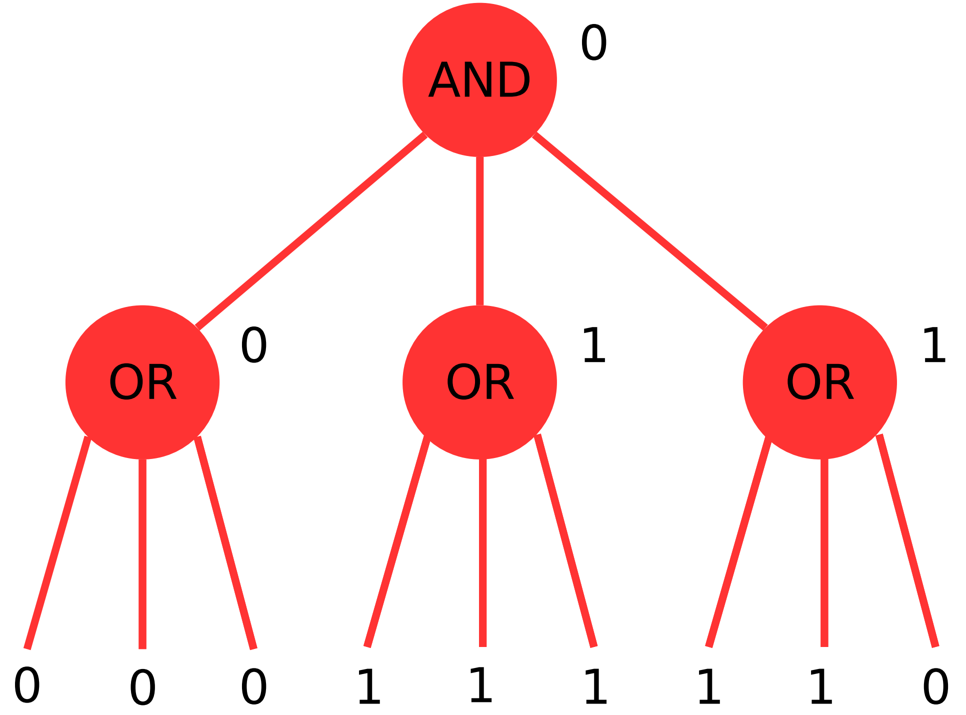
MIN-MAX trees and AND-OR trees

- An AND-OR tree is just a MIN-MAX tree restricted to the values {0,1}!
- So MIN-MAX is at least as hard as AND-OR.

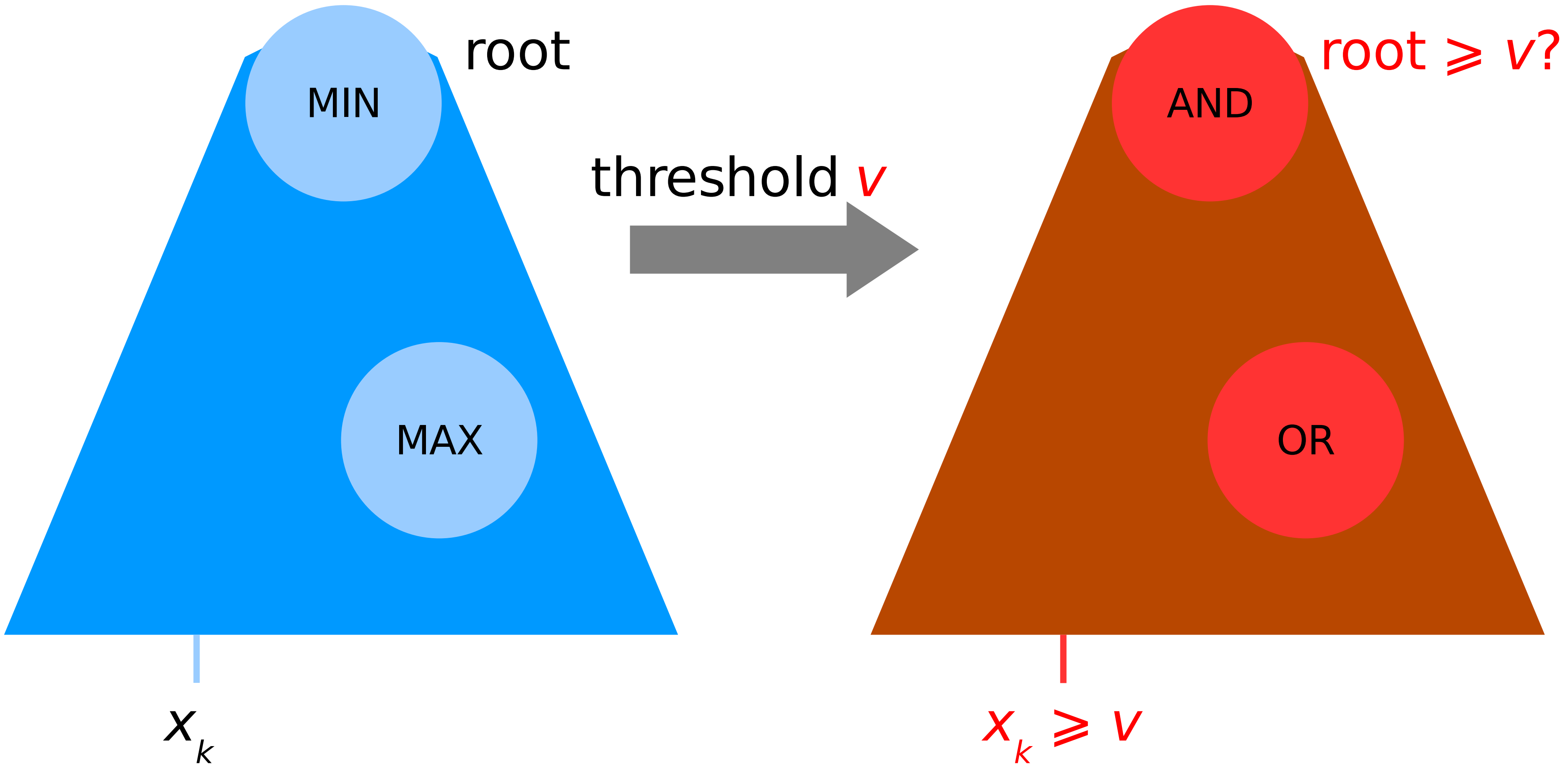


MIN-MAX trees and AND-OR trees

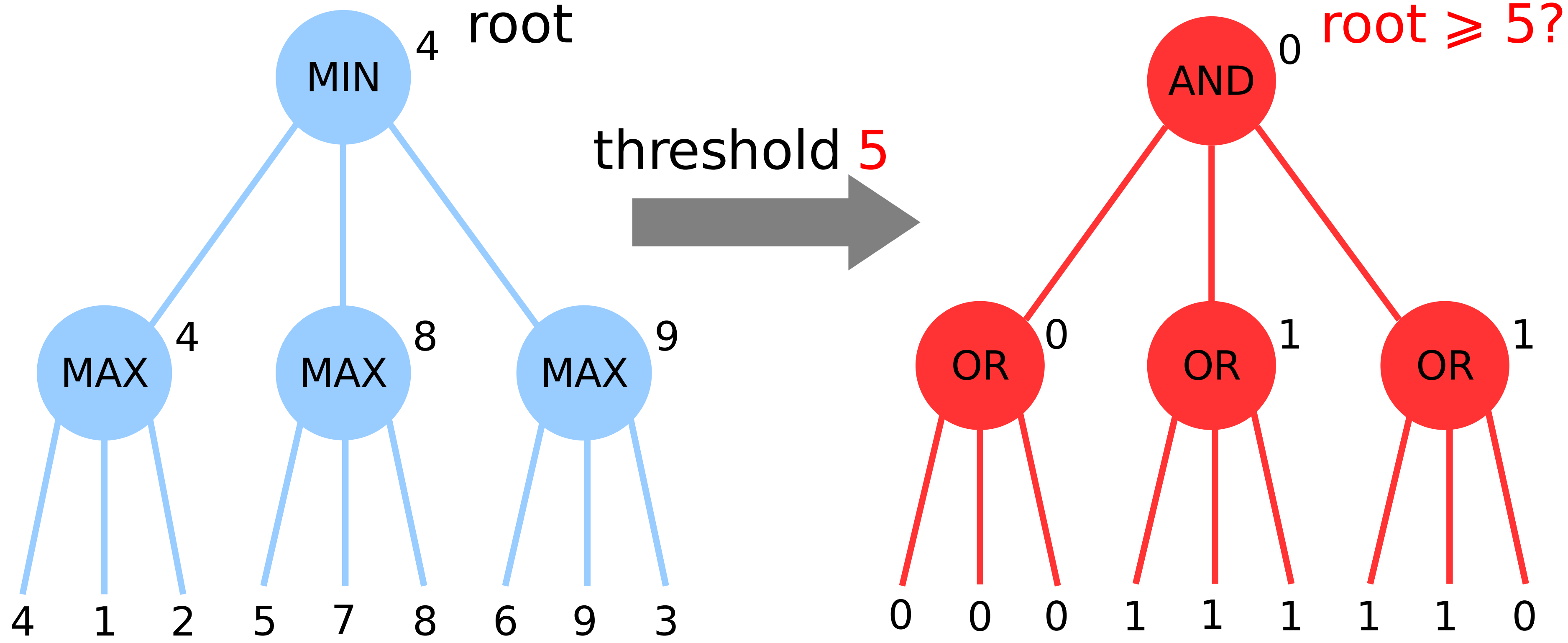
- An AND-OR tree is just a MIN-MAX tree restricted to the values {0,1}!
- So MIN-MAX is at least as hard as AND-OR.



You can also turn MIN-MAX trees into AND-OR trees

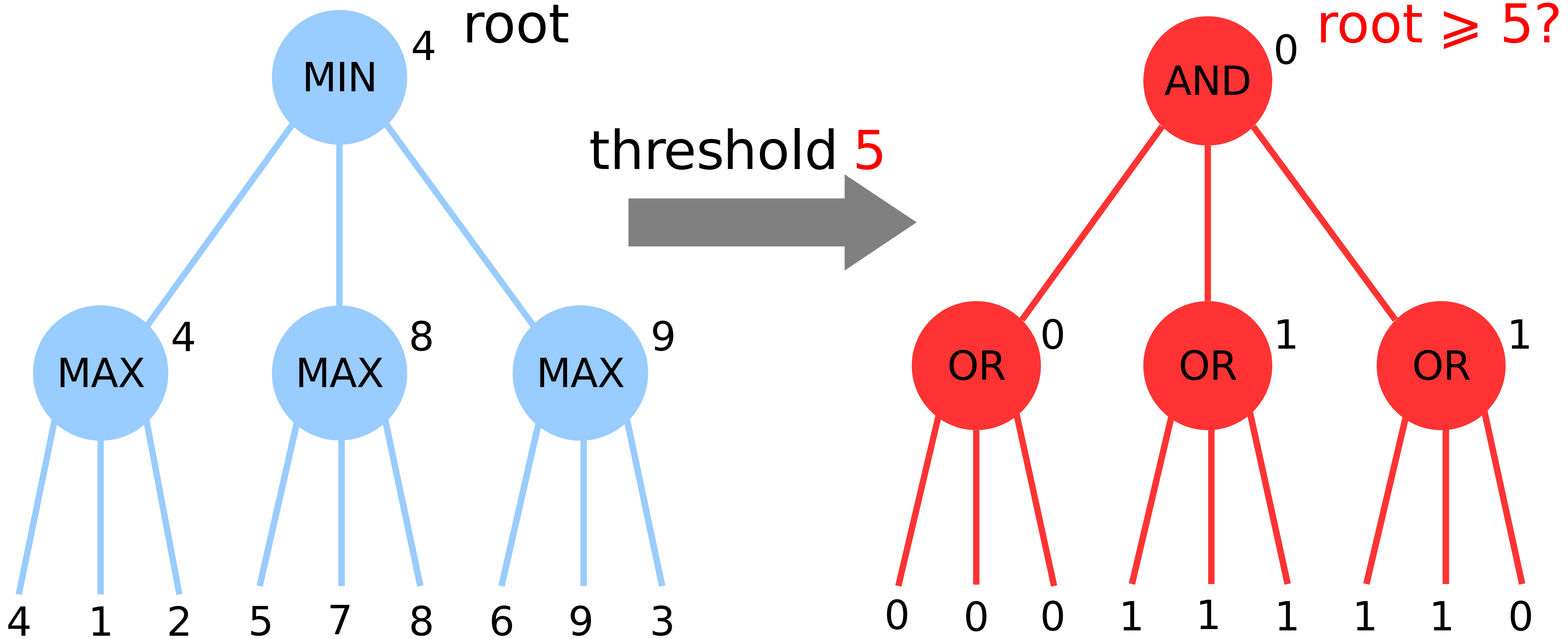


You can also turn MIN-MAX trees into AND-OR trees



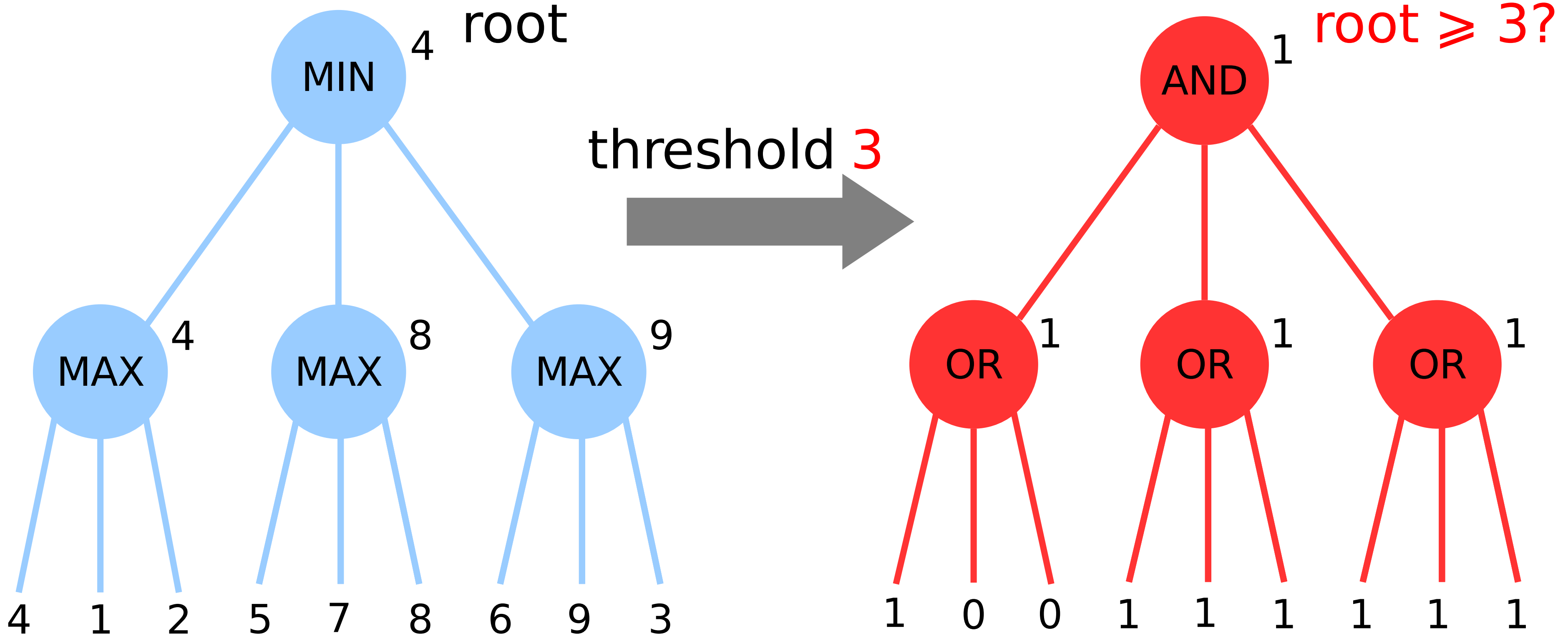
This immediately suggests binary search...

Combining AND-OR and binary search



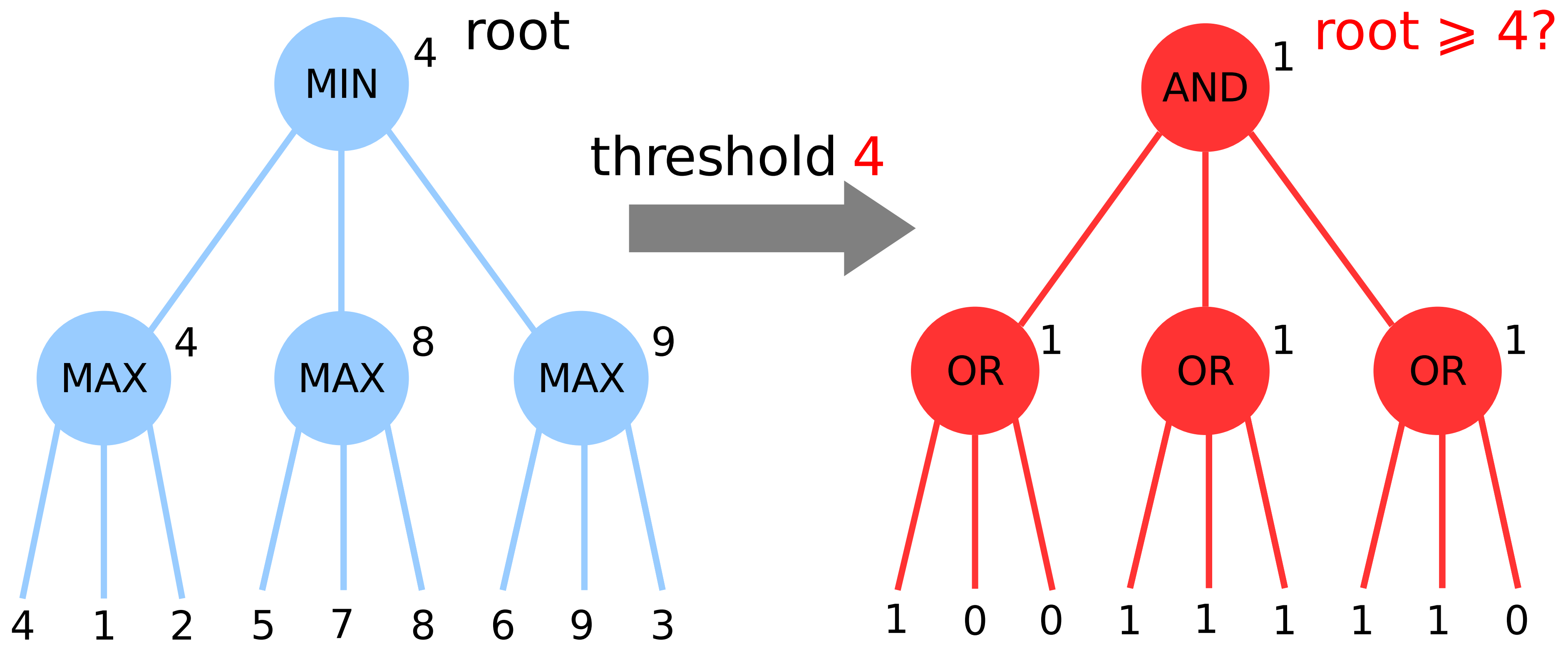
- Is root ⩾ 5? No.

Combining AND-OR and binary search



- Is root ⩾ 5? No.
- Is root ⩾ 3? Yes.

Combining AND-OR and binary search



- Is root ⩾ 5? No.
- Is root ⩾ 3? Yes.
- Is root ⩾ 4? Yes.

} root = 4

Combining AND-OR and binary search

We can consider two models of ordered non-binary data...

- in the input value query model, we have direct access to $x_1,\ldots,x_N$ through a black box;
- in the comparison query model, we are restricted to making comparisons of the form $[x_j < x_k]$.

Problems with combining AND-OR and binary search

We need to find the midpoint of subintervals of the form $[\alpha, \beta]$.

In the comparison query model, the midpoint of an interval cannot be directly computed.

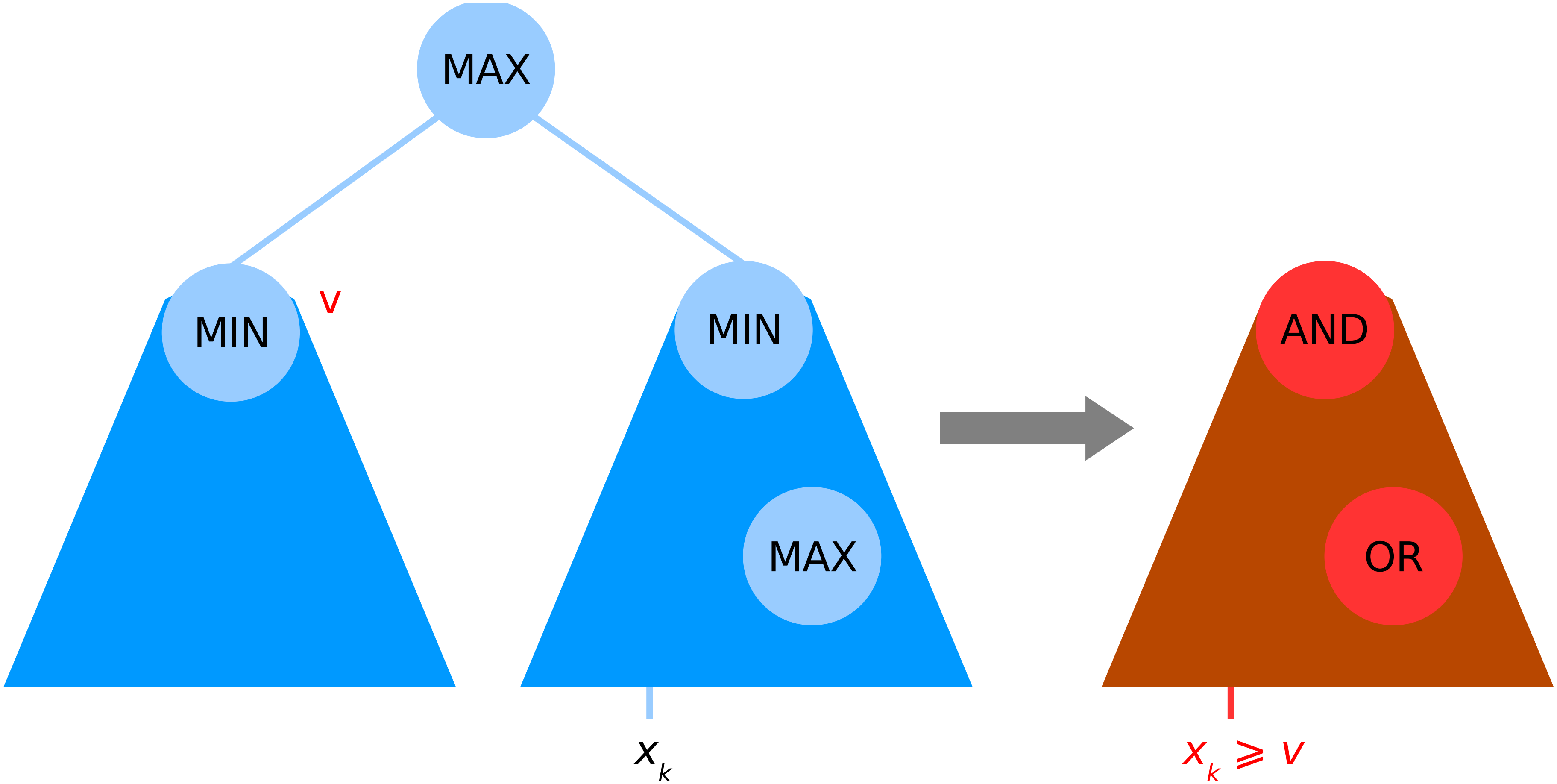
In the input query model, if the numerical range is too large, the binary search may not converge in a logarithmic number of steps.

Saks-Wigderson algorithm

Saks and Wigderson [SW86] showed that...

- the optimal classical randomized algorithm for AND-OR tree evaluation makes $\Theta(N^{0.7537\ldots})$ queries;
- there is an algorithm for MIN-MAX tree evaluation which makes this number of queries, using AND-OR tree evaluation as a subroutine.

Saks-Wigderson algorithm



$$T_N = 3/2\ T_{N/2} + O(N^{0.7537...})$$

This implies a $\Theta(N^{0.7537...})$ algorithm.

Quantum algorithm for AND-OR trees

- There is a lower bound of $\Omega(N^{1/2})$ [BS04]
- There is a "more-or-less" matching algorithm that makes $O(N^{1/2+\varepsilon})$ queries [FGG07, CCJY07, A07+CRŠZ07]

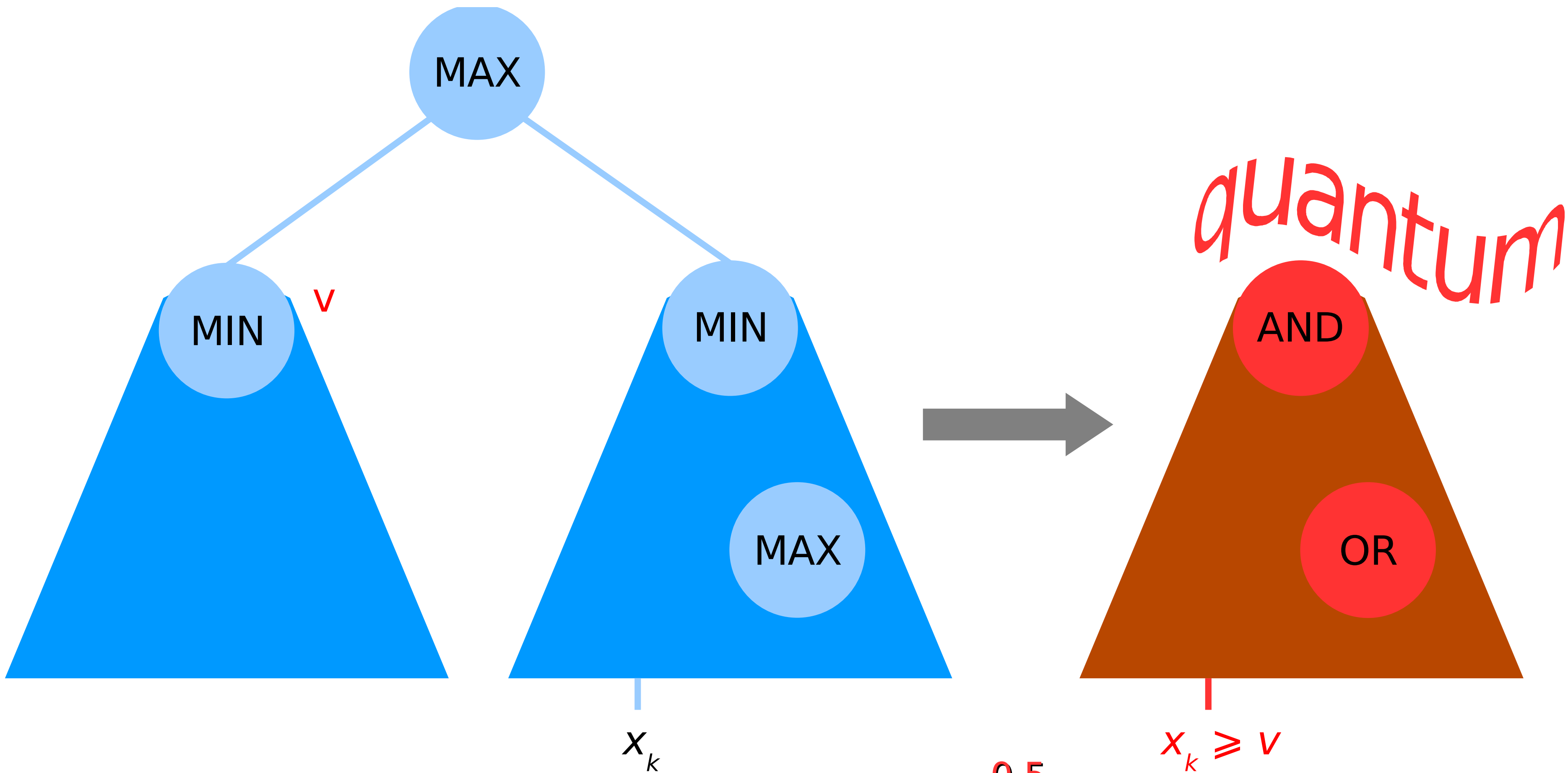
Quantum algorithm for AND-OR trees

- There is a lower bound of $\Omega(N^{1/2})$ [BS04]
- There is a "more-or-less" matching algorithm that makes $O(N^{1/2+\varepsilon})$ queries [FGG07, CCJY07, A07+CRŠZ07]

The "obvious question"...

Do these results generalize to MIN-MAX tree evaluation?

"Quantum Saks-Wigderson"

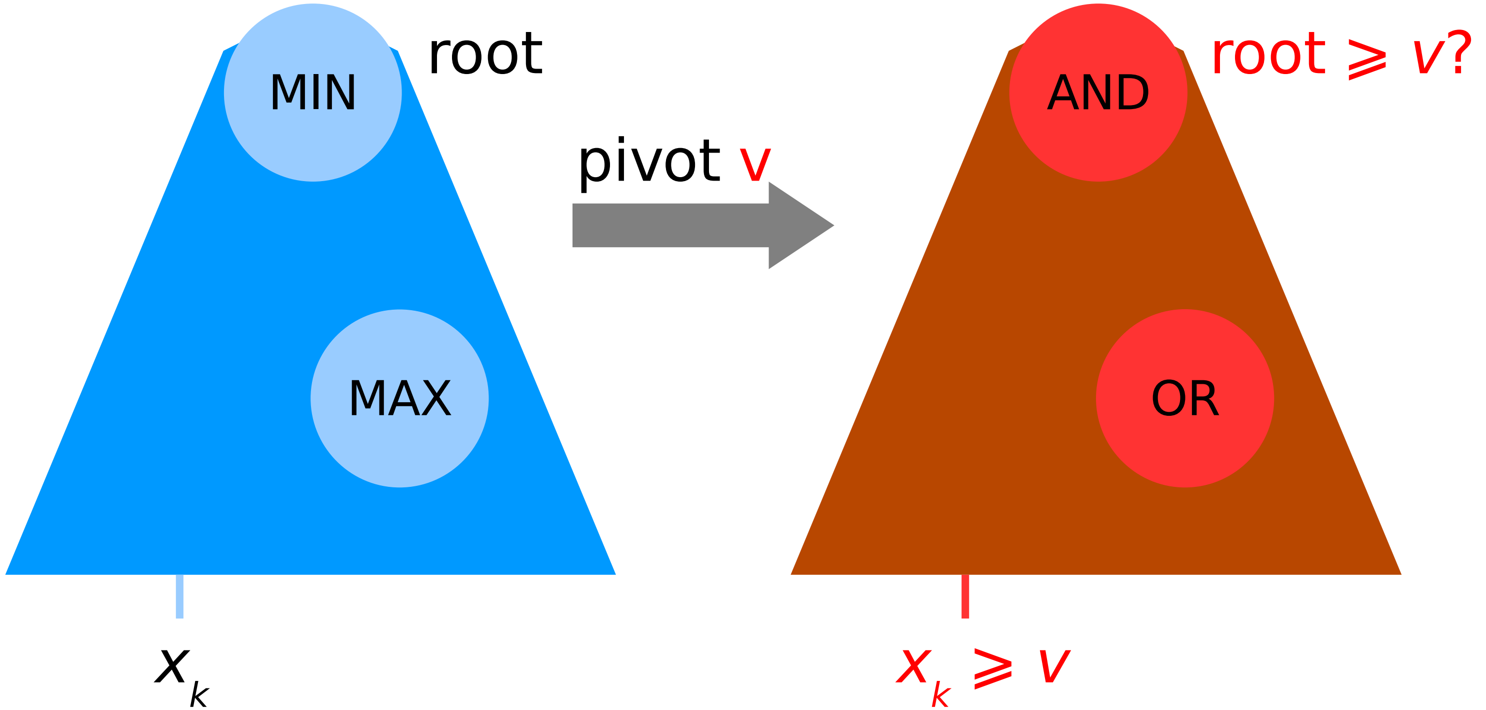


$$T_N = 3/2\ T_{N/2} + O(N^{\cancel{0.7537\ldots}\ 0.5})$$

This implies an $O(N^{0.5850\ldots})$ algorithm.

Can we do better?

- We *could* try to analyze the AND-OR tree algorithm and try to apply it directly to MIN-MAX trees...
- A better idea:
  perform a binary search...

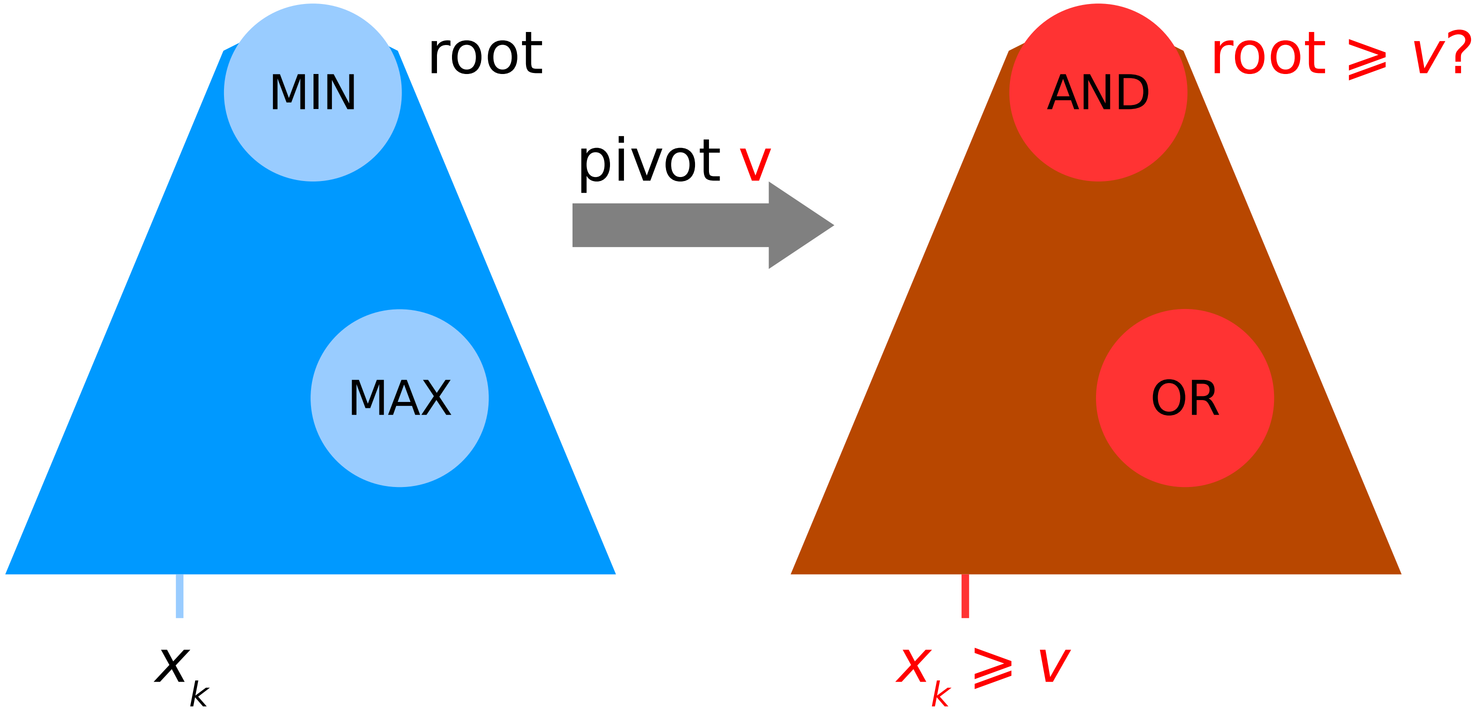


Can we do better?

- We *could* try to analyze and try to apply it direct

But haven't we already established that this approach is full of problems?

- A better idea:
  perform a binary search...



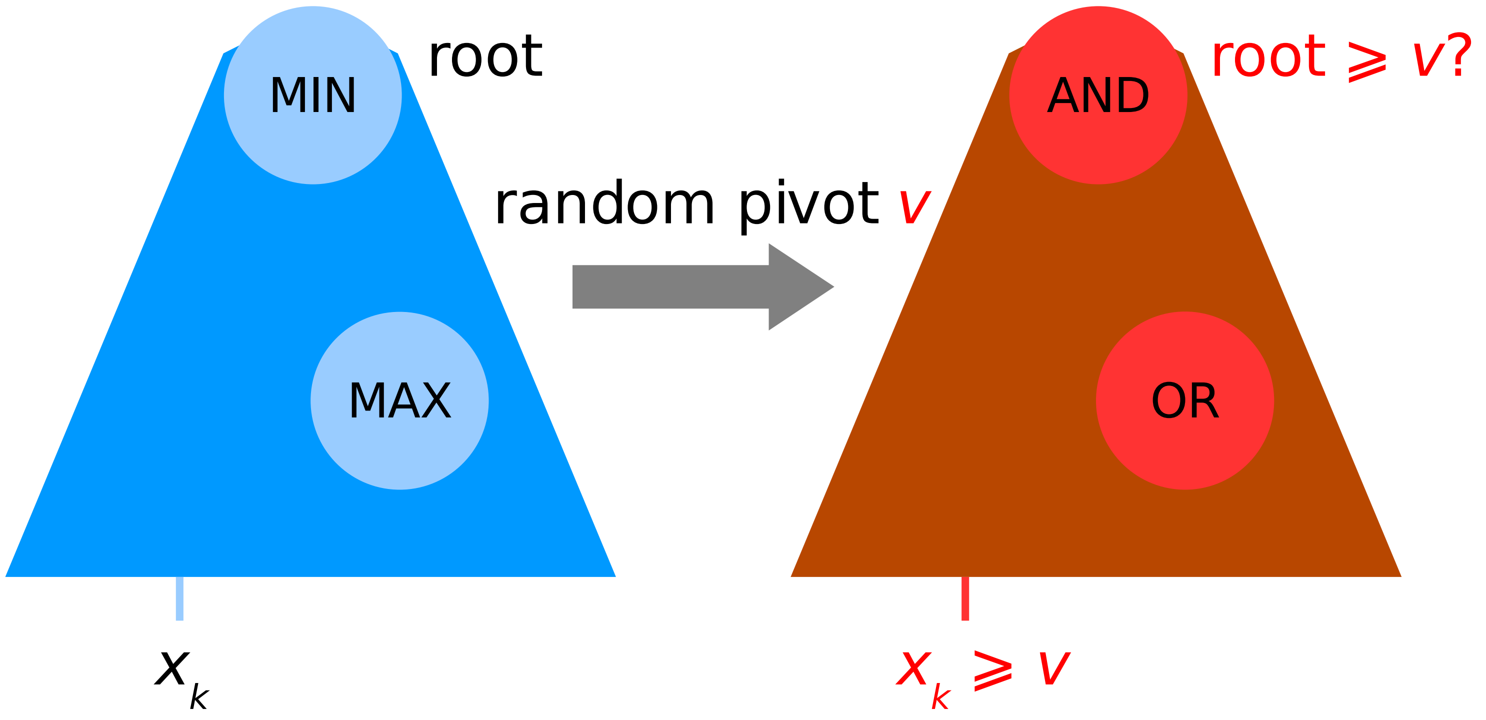
Solution: use random pivots

- A better idea:
  perform a binary search using random pivots.

- Classically, finding a random pivot is as hard as searching, which can take $\Omega(N)$ queries to do even once!
- We have a quantum algorithm to find a pivot with cost $O(\sqrt{N})$: Grover's search!

Quantum algorithm for evaluating MIN-MAX trees

- A better idea:
  perform a binary search using random pivots.



Quantum algorithm for evaluating MIN-MAX trees

- The algorithm runs for O(log N) stages.
- Each stage costs $O(\sqrt{N} \log\log N)$.

To amplify the subroutines to lower the error probability to O(1/log(N))...

Quantum algorithm for evaluating MIN-MAX trees

- The algorithm runs for O(log N) stages.
- Each stage costs O($\sqrt{N}$ ~~loglog N~~).

It turns out that this is unnecessary!
(Using a trick involving a stack...)

Quantum algorithm for evaluating MIN-MAX trees

- The algorithm runs for O(log N) stages.
- Each stage costs $O(\sqrt{N})$.

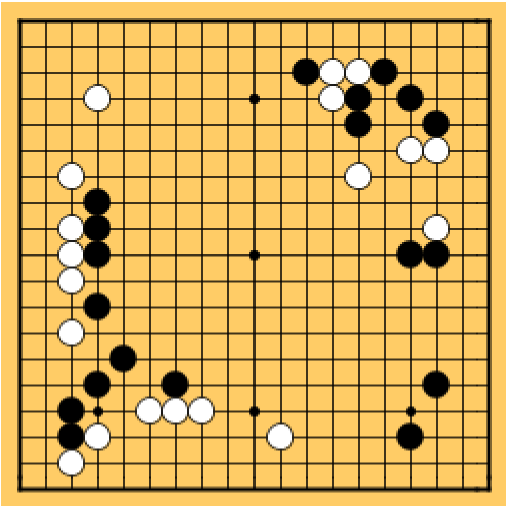
This gives a quantum algorithm for evaluating MIN-MAX trees...

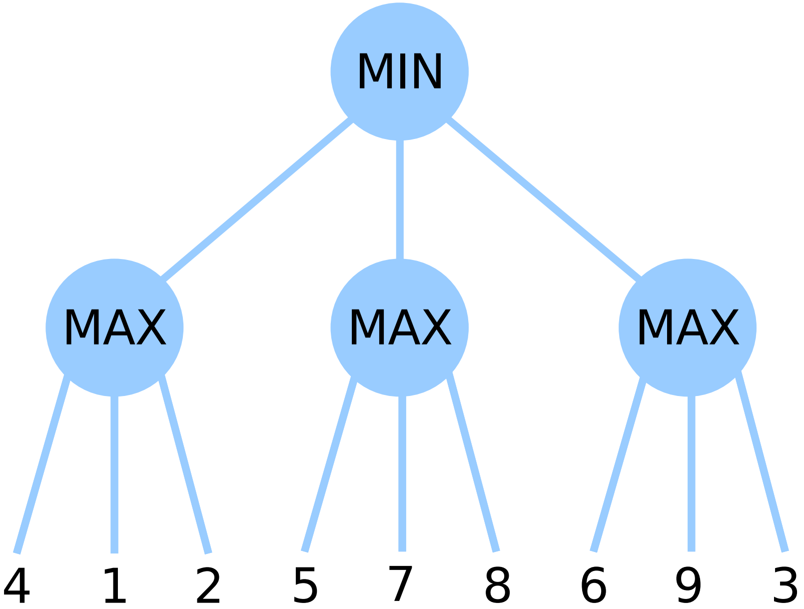
Total cost: $O(\sqrt{N} \log N)$

This is $O(N^{1/2+\varepsilon})$ for an arbitrarily small constant ε.

Obtaining the optimal move

- If the values of the leaves $x_1,\ldots,x_N$ are distinct, this is easy.
- Otherwise, we can use the quantum minimum/maximum finding algorithm [DH96].





Summary

- Classically, the Saks-Wigderson reduction from MIN-MAX to AND-OR uses $\Theta(N^{0.7537\ldots})$ queries.
- Calling the quantum AND-OR subroutine results in an $O(N^{0.5850\ldots})$ algorithm, which is not optimal!
- The classical algorithms are based on examining the subtrees of the tree.

Summary

- Our quantum algorithm performs a binary search using random pivots and requires $O(N^{1/2+\varepsilon})$ queries, which is (close to) optimal.

- Conversely, binary search is too costly for a classical algorithm.
  - The ideas behind the quantum algorithm don't work in the classical setting!

Summary (chart)

Classical: $\Theta(N^{0.7537\ldots})$

- Binary search is too costly
- Based on evaluating subtrees of the MIN-MAX tree
- Doesn't get full speedup from quantum AND-OR subroutine

Quantum: $O(N^{1/2+\varepsilon})$

- Uses binary search
- Based on evaluating the entire tree as an AND-OR tree, with different thresholds
- Gets full speedup from quantum AND-OR and Grover's search subroutines

The Moral of the Story

What works in the classical setting may fail to work in the quantum setting.

What fails to work in the classical setting may work very well in the quantum setting.

To develop quantum algorithms, one must be willing to abandon classical intuitions!

Thanks!

References:

- [CGY07] Quantum Algorithms for Evaluating MIN-MAX Trees.
  arXiv:quant-ph/0710.5794

- [FGG07] A Quantum Algorithm for the Hamiltonian NAND Tree.
  arXiv:quant-ph/0702144

- [A+CRŠZ07] Every NAND formula on N variables can be evaluated in time $O(N^{1/2+\varepsilon})$.
  arXiv:quant-ph/0703015