# Quantum backtracking and tree-search methods

Search a pruned tree of partial assignments while preserving the classical branching task description.

Research 21 min lesson Dr. Charith Jayasekara

**By the end, you'll be able to**

- Identify early rejection in a three-variable constraint tree.
- Separate tree size, depth and local-test cost.
- Explain the coherent-access requirements of backtracking speedups.

## Search a tree of partial answers

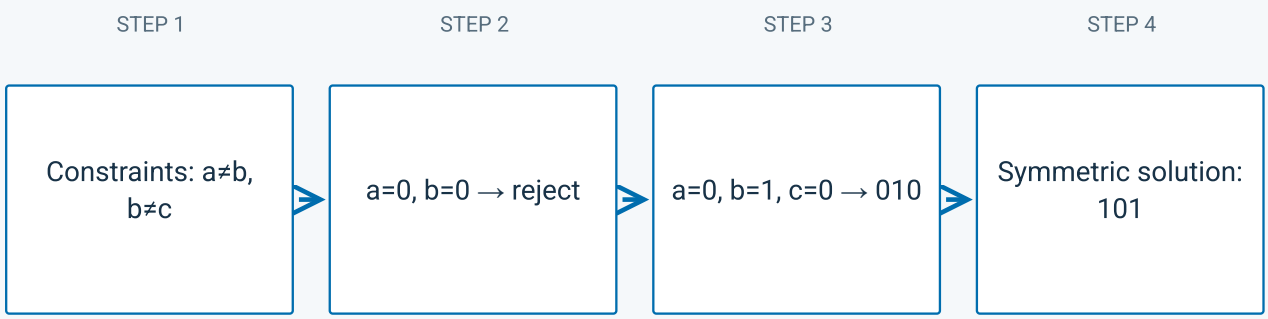
Backtracking solves many constraint problems by building a candidate one choice at a time. A partial assignment is rejected as soon as it violates a constraint. The surviving possibilities form a search tree. Quantum backtracking uses a quantum walk on a suitably represented tree to reduce the number of tests under stated conditions. It exploits the pruned tree, not merely the full space of complete assignments.

That distinction is useful. If classical reasoning prunes almost every branch, a blind Grover search over all complete assignments may throw away valuable structure. A quantum backtracking method aims to preserve access to the classical branching and rejection rules while applying a coherent search primitive. It still needs those rules to be evaluated reversibly and consistently.

Review reversibility, controlled operations and complexity. A tree has a root, parent–child connections and a depth equal to the number of choices along a path.

WORKED DIAGRAM · AA17

### Prune before completing the assignment




The two satisfying assignments are 010 and 101. Branches with a=b can be rejected before c is assigned, so the relevant search tree preserves useful pruning.

## State the tree-access setup

Assume a classical backtracking rule determines which variable to assign next, how to branch and whether a partial assignment is rejected, accepted or still undecided. The tree contains T tested vertices and has a depth bounded by the number of variables n. The quantum task is to find a solution or decide none exists with bounded error, using coherent access to those local rules.

The tree is often implicit: it is defined by the predicates rather than stored as a complete adjacency list. This avoids paying to enumerate every vertex first, but it makes reversible local computation essential. If the classical solver changes its branching rule unpredictably based on an external global history, additional work is needed to define the corresponding coherent tree oracle.

## Work a tiny constraint tree

Take three binary variables a, b and c, with constraints that a differs from b and b differs from c. Branch in the order a, then b, then c. From the root, choose a equal to zero. Setting b to zero violates the first constraint and rejects that branch immediately. Setting b to one survives. Then c equal to zero gives the solution 010, while c equal to one is rejected.

The branch with a equal to one is symmetric and contains solution 101. The tree has useful early pruning: several complete assignments never need to be evaluated. A classical depth-first search follows one route and backtracks after rejection. A quantum walk uses local reflections built from the tree structure, allowing amplitudes associated with different branches to interfere.

The example is too small to show a practical speedup. Its purpose is to identify the local operations the quantum method needs: encode a partial assignment, find its parent and children, test constraints and identify a complete solution. If those operations are unclear, a square-root claim about tree size has no concrete implementation behind it.

The local constraints reveal the two solutions and the early rejection rule.

$$a \oplus b = 1, \qquad b \oplus c = 1 \quad \Longrightarrow \quad a = c = 1 - b, \qquad (a, b, c) \in \{(0,1,0), (1,0,1)\}.$$

All variables are binary. A partial branch with $a = b$ can be rejected before choosing $c$, which explains the useful pruning in the chapter's tree. This is an instance calculation, rather than a runtime theorem for arbitrary trees.

## What the theorem improves

Montanaro's quantum backtracking method gives a bounded-error speedup in the number of local tests, with a bound involving the square root of tree size and additional depth-related factors. The original result includes a dependence on the number of variables, so "replace T by square root T" is an incomplete summary. Read the quantum backtracking paper.

The theorem is conditional on access to the relevant tree and tests. It does not prove that every industrial constraint solver becomes faster by the same factor. Real solvers use learning, restarts, propagation, caching and data structures that change the search dynamically. A useful application analysis identifies which of those techniques can be represented in the chosen quantum model.

## Follow the construction conceptually

Define the root and the branching rule. Give each partial assignment a reversible encoding and provide coherent parent, child and predicate operations. Build the local reflections used by the tree walk. Apply the search or detection procedure with its required size and depth bounds, then extract a candidate path and verify the complete assignment classically.

Some algorithms separate detecting whether a solution exists from locating one. Recovering an explicit solution can require additional searches or path-finding steps. Count those operations rather than treating a detection theorem as a complete solution-printing procedure. If the tree size is unknown, the schedule must also handle that uncertainty.

## Why pruning and depth both matter

A large shallow tree and a long narrow tree can have the same number of vertices but different walk behaviour and implementation costs. Depth bounds affect spectral analysis and the size of partial-assignment registers. The branching factor changes local state preparation. These parameters belong beside T in a resource estimate.

Classical pruning can reduce T dramatically, and improving the classical predicate can help the quantum method too. But a pruning rule that is cheap classically may be costly to compute reversibly with limited workspace. The algorithmic comparison should include the cost per local test, not just the number of tests. Memory cleanup and reversible constraint propagation can dominate a circuit.

## Compare against the right classical solver

Start with an established classical baseline and record its actual branching strategy, solution quality and runtime distribution. Translate a well-defined subset of that strategy into a coherent tree model. Estimate logical gates, workspace, success repetitions and fault-tolerant costs. Compare on a family of instances rather than selecting only cases where the classical tree is unusually large.

If a problem has a polynomial-time classical method, searching an exponential backtracking tree may be the wrong formulation entirely. Even for hard worst-case problems, practical instances can have exploitable structure. The existence of a quantum tree-search theorem does not remove the need for problem-specific algorithm design.

## Practice with answers

**Exercise 1.** Which two assignments satisfy the worked constraints, and which partial assignment can be rejected before choosing c?

> Check the worked answer
>
> The solutions are 010 and 101. Any partial assignment with a equal to b can be rejected immediately, before c is considered. That early rejection is the pruning structure the tree representation preserves.

**Exercise 2.** Why is a classical solver's runtime not automatically a valid T for a quantum backtracking theorem?

> Check the worked answer
>
> T counts vertices or tests in a specified tree model. Runtime also includes data structures and changing heuristics. The theorem needs coherent local access to a well-defined tree, and the cost of implementing each test may differ substantially from its classical cost.

## Check your understanding

PAUSE & THINK

### Which partial assignment can be rejected before choosing c?

- [ ] a=0,b=1
- [ ] a=1,b=0
- [ ] a=0,b=0

---

**Answer:** a=0,b=0. The constraint a≠b already fails for a=0,b=0. A coherent tree oracle must implement that local rejection rule reversibly.

## Sources

- Read the quantum backtracking paper