# Crate branch_and_bound



This library implements generic branch-and-bound and backtracking solver.

Branch-and-bound (and backtracking, which is its special case) is the method of solving an optimization problem by recursively breaking a problem down to subproblems and then solving them. Unlike brute-force, branch-and-bound will discard a subproblem if it discovers that the best potentially obtainable solution to this subproblem is not better than the current best solution (aka incumbent).

To use the library, one shell implement a type that represents a problem (subproblem) and implement the `Subproblem` trait for it.

One can then `solve` an instance of problem using one of the predefined methods (DFS, BFS, BeFS, etc) or use `solve_with_container`, through which custom strategies can be implemented.

## Re-exports

```
pub use bnb_aware_containers::BnbAwareContainer;
```

## Modules

bnb_aware_containers

## Enums

| | |
|---|---|
| SubproblemResolution | Represents the set of subproblems of an intermediate problem or the value of the objective function of a feasible solution (leaf node). |
| TraverseMethod | Order of traversing the subproblem tree with `solve`. See variants' docs for details. |

## Traits

| | |
|---|---|
| Subproblem | A problem (subproblem) to be solved with branch-and-bound |

## Functions

| | |
|---|---|
| solve | Solve a problem with branch-and-bound / backtracking, using one of the default strategies. |
| solve_with_container | Solve a problem with branch-and-bound / backtracking using a custom subproblem container with a custom strategy. |