Framework for Branch and Bound Applications

API and Infrastructure Design:

Solution - Abstract Class: The application programmers need to extend this class to represent the application-specific partial cost for a node in the search tree. For instance, in the case of Traveling Salesman Problem, (TSP) the implementation class would contain the information about the partial tour (from the root to the current node)

BranchAndBound Class: This class represents the task involved in solving any combinatorial optimization problem that uses the branch-and-bound technique. This extends Task class whose objects represent a node in the search tree. This task represents atomic tasks, decomposable tasks (those that can be split into multiple sub-tasks) and successor tasks (that compose the results of the sub-tasks). The execute method either

1. constructs smaller BranchAndBound tasks that correspond to its children, or
2. executes the atomic task by fully searching a subtree, returning the best solution

Shared - Interface: This represents the shared object that stores the upper bound on the cost of a minimal solution. All tasks have access to this shared object. To propagate a newer shared object to other tasks, the task invokes its setShared method which calls the executing computer's setShared method. This, in turn propagates the proposed newer shared object to the Space by calling the Space's setShared method (using a Space Proxy), only if the proposed shared object is newer than the computer's existing shared object. Similarly, the Space propagates the received shared object to all the computers except the one from which it received the shared object, only if it is newer than the Space's shared object.
For TSP, the initial value for the Shared object is computed at the client side using a greedy approach and propagated to the Space and Computers.

Addressing Design issues:

The framework allows the application programmers to focus on the problem-specific aspects of branch and bound and not on the other aspects like branching, exploring sub-trees to find the minimum cost, etc. For instance, the framework provides a BranchAndBound class (as described above) which can be used to model many optimization problems such as TSP, Integer Programming, Minimum Spanning Tree. In other words, the BranchAndBound class represents the task involved in solving any combinatorial optimization problem.

This framework doesn't stop users from running other applications like computing fibonacci sum, visualizing a MandelbrotSet, etc. These applications can make use of the respecting Task classes (FibonacciTask and MandelbrotSetTask). This is achieved by having a separate abstract class called "Task" which can be extended by application-specific classes.

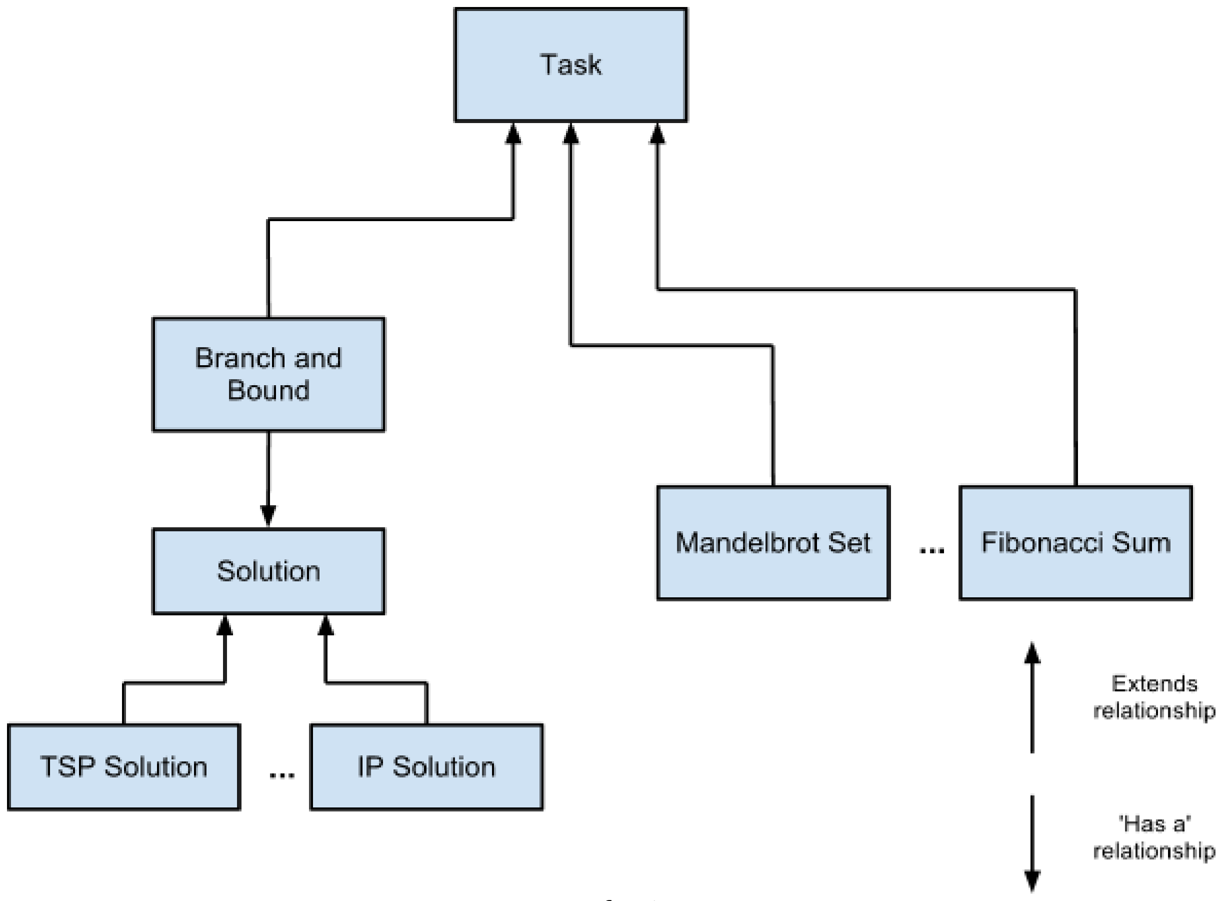

fig. 1

Figure 1 shows the design of the classes and explains how the framework can support multiple combinatorial optimization problems (eg. TSP, Integer Programming, etc.) as well as other applications like Mandelbrot Set visualization and Fibonacci Sum.