Algorithm 1 Improvement-Based Bee Colony Optimization algorithm

```
procedure BCOi(vessels, B, NC, RunTime)
    while SessionTime() ≤ RunTime do
        BeeSol ← InitialSolution(B, vessels)
        for i ← 1, NC do
            BeeSol ← FirstTransformation(B)
            bestBee ← SmallestCost(B)
            BeeSol ← SecondTransformation(B)
            BeeSol ← RecruitingProcess(B)
        end for
        currentBest ← SmallestCost(B)
        Update(GlobalBest)
        currentBest ← Improve(GlobalBest ∨ currentBest)
        Update(GlobalBest)
    end while
end procedure
```

Algorithm 2 Generating initial solutions for BCOi

```
procedure InitialSolution(B, vessels)
    i ← 1
    while i ≤ B do
        UnusedVessels ← {1, 2, ..., l}
        BeeSol(i) ← {}
        Ψ(i) ← startΨ
        while UnusedVessels ≠ ∅ do
            v ← Roulette(UnusedVessels, selectionCriteria)
            ξ ← ψ(i, v)
            pos ← RandomPosition(v, ξ)
            Update(BeeSol(i), pos)
            Reduce(ψ(i), pos)
            UnusedVessels ← UnusedVessels\{v}
        end while
        i ← i + 1
    end while
    BeeSol ← RecruitingProcess(B)
    Update(GlobalBest)
end procedure
```

Algorithm 3 BCOi first solution transformation

```
procedure FirstTransformation(B)
    for i ∈ B do
        V_b ← FindVessel(ψ(i))
        s_b ← SelectRandomVessel(V_b)
        for v ∈ s_b do
            if PossibleToMove(v) then
                ξ ← ψ(i, v)
                pos ← RandomBetterPosition(v, ξ)
                Update(BeeSol(i), pos)
            end if
        end for
    end for
end procedure
```

Algorithm 4 BCOi second solution transformation

```
procedure SecondTransformation(B)
    for i ∈ B do
        Update(ψ(i), BeeSol(i))
        V_b ← FindVessel(ψ(i))
        s_b ← SelectRandomVessel(V_b)
        for v ∈ s_b do
            if PossibleToMove(v) then
                ξ ← ψ(i, v)
                pos ← RandomBetterPosition(v, ξ)
                Update(BeeSol(i), pos)
                Reduce(ψ(i), pos)
            end if
        end for
    end for
end procedure
```

Algorithm 5 BCOi improvement phase

```
procedure Improve(sol)
    repeat
        Solution ← {}
        Sort(vessels)
        temp ← sol
        for i ∈ vessels do
            V ← ConflictVessels(i)
            for all V_j do
                movingVessels ← V_j ⋃ {i}
                ξ(V_j) ← startΨ(V_j)
                tempAllocation(movingVessels) ← Solve(movingVessels, Ψ)
            end for
            MaxSavings(tempAllocation)
            Update(Solution)
            if Solution ≠ {} then
                Update(sol)
                Break()
            end if
        end for
    until temp = sol
end procedure
```