```
BipartiteMatchingOptimize( Solution S )
    Rideshares = S.Drivers()
    last_score = 0.0
    while( score(S) > last_score )
        last_score = score(S)
        CompatibleEdges = unmatchedEdges(Rideshares, S.Riders)
        while( S.hasAugmentingPath() )
            augmentingPath = GetAugmentingPath(S)
            for each edge in augmentingPath do
                if( edge = matched ) edge = unmatched
                if( edge = unmatched ) edge = matched
        end while
        for each edge in Edges.matchedEdges() do
            add edge ->rider to edge->rideshare
            remove edge->rider from S.Riders
    end while
    return S

GetAugmentingPath( Solution  S )
    for each rider in UnmatchedRiders(S) do
        traverse unmatched edges from rider to rideshares
        traverse matched edges from rideshares to riders
        return augmentingPath if an unmatched rideshare is reached
    return false
```