Your algorithm is just fine if you embedded it into a backtracking algorithm (http://en.wikipedia.org/wiki/Backtracking). The easiest way to do it is to call your method recursively once you found a matching pair/chow/... but discard current choice it if not. In pseudo code this looks something like this:

```
1. Mark all tiles as nonwinning
2. Call function Backtracking

Function Backtracking:
1. If all tiles are marked winning return WINNING else NONWINNING
2. Visit tiles as described in your algorithm
3. When found a chow/pong/... or the first pair
   3.1. Mark those tiles as winning.
   3.2. Afterwards call method Backtracking.
   3.3. If return value of 3.2 is WINNING also return WINNING
   3.4. Else unmark the tiles as nonwinning again
4. If not finished search for pair/chow/... by looping to 3.
5. All combinations are done and no winning movewas found so return NONWINNING
```

The idea behind is that you first try each pair/... as part of a winning hand but if this does not work just try the same by assuming it is not part of the winning hand.

If you are not only interested if it is a winning hand but all possible combinations of winning pairs/chows/... skip the return in step 3.3.