Algorithm TPL-refinement (q, S_{cnd}, P_{rfn}, N_{rfn})

1. for each point p in S_{cnd}
2. for each other point $p' \neq p$ in S_{cnd}
3. if $dist(p,p') < dist(p,q)$
4. $S_{cnd} = S_{cnd} - \{p\}$; goto 1
5. if p is not eliminated initialize $toVisit(p) = \emptyset$
6. repeat
7. **refinement_round**(q, S_{cnd}, P_{rfn}, N_{rfn})
8. if ($S_{cnd} = \emptyset$) return // terminate
9. $P_{rfn} = N_{rfn} = \emptyset$ //initialization of next round
10. Let N be the lowest level node that appears in the largest number of sets $toVisit(p)$, where $p \in S_{cnd}$
11. remove N from all $toVisit(p)$ and access N
12. if N is a leaf node
13. $P_{rfn} = \{p / p \in N\}$ //P_{rfn} contains only the points of N
14. if N is an intermediate node
15. $N_{rfn} = \{N' / N' \in N\}$ //N_{rfn} contains the child nodes of N

End TPL-refinement

Algorithm refinement_round(q, S_{cnd}, P_{rfn}, N_{rfn})

1. for each point p in S_{cnd}
2. for each point p' in P_{rfn}
3. if $dist(p,p') < dist(p,q)$
4. $S_{cnd} = S_{cnd} - \{p\}$ //false hit
5. goto 1 //test next candidate
6. for each node MBR N in N_{rfn}
7. if $minmaxdist(p,N) < dist(p,q)$
8. $S_{cnd} = S_{cnd} - \{p\}$ //false hit
9. goto 1 //test next candidate
10. for each node MBR N in N_{rfn}
11. if $mindist(p,N) < dist(p,q)$ add N in $toVisit(p)$
12. if ($toVisit(p) = \emptyset$)
13. $S_{cnd} = S_{cnd} - \{p\}$ and report p // actual result

End refinement_round