

```

// store candidate nodes for every unrolled iteration
1: C = array(PAT_SIZE, UNROLL, MAX_DEGREE);
2: Csize = array(PAT_SIZE, UNROLL);
3: iter = array(PAT_SIZE);
   // iterate for unrolled iterations
4: uiter = array(PAT_SIZE);
5: l = 0;
6: while (true) {
7:     if (l < Q.size) {
8:         // if in the first of the unrolled iterations
           // and if candidate set is empty
8:         if (uiter[l] == 0 && Csize[l][0] == 0) {
9:             // extend subgraphs for all unrolled iterations
           getCandidates(G, Q, l, C, Csize, UNROLL);
           // if no subgraph can be extended
10:         if (l == 0 && Csize[0][0] == 0) {
           // try to steal from other warps
11:             if (!local_steal()) {
12:                 if (!global_steal()) { break; }}}
13:         iter[l] = 0; uiter[l] = 0; }
           // if there are more unrolled iterations
14:         if (uiter[l] < UNROLL) {
           // and if there are unexplored nodes in current
           // unrolled iteration, go to next level
15:         if (iter[l] < Csize[l][uiter[l]]) { l++; }
16:         else {
           // if all candidates are explored in current
           // unrolled iteration, go to next unrolled iteration
17:         Csize[l][uiter[l]] = 0;
18:         iter[l] = 0;
19:         uiter[l]++; } }
20:         else {
           // if all unrolled iterations have been executed
           // reset unroll iterate
21:         uiter[l] = 0;
           // and backtrack to previous level
22:         if (l > 0) { l--; iter[l] += UNROLL; } } }
23:     else {
24:         for (i = 0; i < UNROLL; i++) Output(C, i, iter);
25:         l--; } }

```