1. Give a 2-approximation algorithm for the vertex cover problem. Explain why it works and why the algorithm provides a solution no worse than 2 times the optimal vertex cover size.
2. Let *VC* be the vertex cover for a graph *G*.
3. Go to each edge, *e,* in the graph with vertices *(u, v)*.
4. If *e* is uncovered, add both vertices *u* and *v* of *e* to *VC.*

We can see that this will work if take into account matchings. Each edge that is uncovered of which we are adding to the vertex cover set can also be added to a matching since it is uncovered by any of the previous vertices in the vertex cover. A maximal matching is one in which an edge cannot be added that is adjacent to another edge. We can see from this definition that if we have a maximal matching that the vertices on the edges in that matching will make up a vertex cover for the graph. If we can see this, then we may also see the fact that we need at least one of the vertices for each edge within the graph to be in the vertex cover per the definition of a vertex cover. Thus, the size of the minimum vertex cover must be at least the size of the maximal matching, and the vertex cover we have found is exactly twice that size. In conclusion, the vertex cover we have found cannot be more than twice the size of the minimum vertex cover, and we can declare to have found a 2-approximation algorithm for vertex cover.

1. Briefly argue whether the 3-Hitting Set problem is fixed-parameter tractable. 3-Hitting Set: Input: A collection *C* of subsets of size three of a finite set *S* and a nonnegative integer *k*. Task: Find a subset *S’ ⊂ S* with *|S’| ≤ k* such that *S’* contains at least one element from each subset in *C*.

The 3-Hitting Set problem is fixed parameter tractable if we give it a parameter *k*, such that a subset *S’* of *S* with cardinality *<= |k|* will contain an element from every subset within *C* will be a 3-hitting set. *C,* in this case, is a collection of subsets of the finite set *S.* This problem is fixed parameter tractable since it has a kernelization. We can see the kernelization if we imagine the hypergraph of *C*. The hypergraph of *C* from above will be similar to a regular graph except that each edge set will be three vertices instead of two. The edges will each be one of the subsets within the collection *C*. We just need to solve this hypergraph for its vertex cover to solve the 3-hitting set problem. One easy step to a kernelization that we can see involves counting techniques. We can see that if a vertex *v* within a hypergraph *H* of *C* has more than *k* edges whose pairwise intersection is *{v}*, we need to add this vertex to the 3-hitting set and decrement *k*. This is very similar to the counting techniques used in the kernelization of vertex cover for a normal graph. We can then take the knowledge that finding a kernelization for a problem means that the problem has a fixed-parameter tractable algorithm for its solution.

1. Recall that we have often considered parameterized version of the Vertex Cover problem using the size of the vertex cover set as our parameter k. On what types of graphs might this parameter not be ideal? Why?

The parameterized version of the vertex cover problem is not as ideal on graphs that have a fairly high vertex cover. For example, if we have a vertex cover of cardinality greater than 190, then we have exceeded the reasonable amount of time metric according to the vertex cover’s klam value. In general non-planar graphs will have a less efficient solution than planar graphs when using the parameterized version of vertex cover. As the vertex cover cardinality grows the fixed parameter tractable algorithm for finding vertex cover also grows exponentially. If a graph is non-planar we can reduce the problem of vertex cover to planar graphs, which means that non-planar vertex cover is more complex than vertex cover on planar graphs.

1. Show that in any Boolean formula in CNF at least half of the clauses can always be satisfied.