```
def mc_rec(self, r, p, x, res):
    """
    Determines all the cliques inside the graph
    :param r: the temporary result (list of int)
    :param p: the set of possible candidates (list of int)
    :param x: the excluded set (list of int)
    :param res: the set of results (list of list of int)
    :return:
    """
    if len(p) == 0 and len(x) == 0:
        res += [r]
    else:
        p1 = p.copy()
        for v in p:
            p2 = [val for val in self._graph.parseNOut(v) if val in p1]
            x2 = [val for val in self._graph.parseNOut(v) if val in x]
            self.mc_rec(r + [v], p2, x2, res)
            p1.pop(p1.index(v))
            x += [v]

def max_clique(self):
    """
    Gets all the cliques from the graph and takes one of the the maximum ones
    :return: the maximum clique of the undirected graph
    """
    mx = 0
    res = []
    clique = []
    self.mc_rec([], self._graph.parseX(), [], res)
    for c in res:
        if len(c) > mx:
            mx = len(c)
            clique = c.copy()
    return clique
```