ASSIGNMENT 3 – ARTIFICIAL INTELLIGENCE (CSL333)

Nipun Shrivastava **Deepak Koli**

2011CS50288 **2011CS50278**

Our algorithm for translation into SAT comprised of 5 set of CNFs.

- CNF 1 was used to ensure that at least a single mapping exist from smaller graph to larger graph.
- CNF 2 set ensured single mapping existed from larger graph to smaller graph.
- CNF 3 and 4 sets made sure that for a pair of nodes in the smaller graph if an edge doesn't exist and the two nodes are mapped to a certain pair of nodes in the larger graph, then no edge should exist between the nodes comprising the pair from the larger graph. Similarly for the case where an edge exist..

Let us describe the clauses in more detail. Let **V(i,j)** denote the literal for a mapping from node i of smaller graph to node j of larger graph. A true for V(i,j) means that if we all the clauses were satisfied then in the sub graph of larger graph which is mapped to the given smaller graph has a mapping from node i of smaller graph to node j of larger graph. For depicting them as integers we have used the following formula (**assuming i and j start from 1 and m is the number of nodes in the larger graph and n is the number of nodes in the smaller graph**):

$$V(i,j) = m*(i-1) + j$$

This gives us a mapping of all V(i,j)'s from 1 to m*n. To get back the information of mapping from the integer (say z) we simply decode it using the following formulae:

$$j = ((z-1) \% m)+1 \qquad i = ((z-j) / m) + 1$$

- **CNF 1:** (1 or 2 or 3 or 4 or … or m) and (m+1 or m+2 or m+3 or m+4 or … or m+m) and … ((n-1)*m+1 or (n-1)*m+2 or (n-1)*m+3 or (n-1)*m+4 or … or (n-1)*m+m)
- **CNF 2:** And of all nC2 pairs of (V(k,i) or V(l,i)) when iterated over k and l. In the outer loop we will run a loop to cover all the m i's..
- **CNF 3:** Edge(i,j) and V(i,k) and V(j,l) -> Edgel(k,l)
  Outer loops picks up all possible pairs of i,j from smaller graph while the inner loops picks up all pairs of k,l from larger graph. In all there will be nC2 * mC2 iterations.
- **CNF 4:** NOT(Edge(i,j)) and V(i,k) and V(j,l) -> NOT(Edgel(k,l))
- Outer loops picks up all possible pairs of i,j from smaller graph while the inner loops picks up all pairs of k,l from larger graph. In all there will be nC2 * mC2 iterations.

  *Edge(i,j) is a function that returns bool value telling if an edge exists or not in smaller graph.
  *Edgel(i,j) is a function that returns bool value telling if an edge exists or not in larger graph.

- Steps to improve performances:

- The first performance improvement we added was using the property of in-degree and out-degree of nodes in the two graph. If the in-degree or the out-degree of the node from larger graph was smaller than the in-degree or the out-degree of the node from smaller graph respectively. Then we don't consider the variable that denotes the mapping of those two nodes. Again this reduces the size of problem for SAT and hence we gain on the time forefront.

- The second performance improvement we added was using the symmetry condition in the final two CNF set.

- The performance improvement using the degree concept reduced the number of literals drastically thus improving the performance of MINISAT.the in-degree or the out-degree of the node from smaller graph respectively. Then we don't consider the variable that denotes the mapping of those two nodes. Again this reduces the size of problem for SAT and hence we gain on the time forefront.