CSE505 : COMPUTING WITH LOGIC
HW 4

Team Members :
Murali Krishna Somanur Ramasamy (***SBU ID: 109769583***)
Sai Santhosh Vaidyam Anandan (***SBU ID: 109819365***)
Sowmiya Narayan Srinath (***SBU ID: 109747156***)

Abductive Logic Programming:

Given a partial program and a set of goals, abductive reasoning allows us to come up with other predicates of the program that would make the goal(s) true. In formal terms, given a program P and goal G, we should determine P' such that (P U P') |= G. Unlike deductive reasoning where goals are inferred from predicates, abductive reasoning allows us to build the facts when some conclusions are known.

Consider the following example program (*Source: Wikipedia*):

```
citizen(X) :- bornInUSA(X).
citizen(X) :- not(bornInUSA(X)), resident(X), naturalized(X).
citizen(X) :- not(bornInUSA(X)), mother(X,Y), citizen(Y), registered(X).
```

Now assume we know that `mother(John,Mary)` and `citizen(Mary)` are true.
Using abductive reasoning we can conclude that either of the following is true if John is a citizen:

1. `bornInUSA(John)`
2. `not(bornInUSA(John)) and resident(John) and naturalized(John)`
3. `not(bornInUSA(John)) and citizen(Mary) and registered(John)`

Design and Modifications:

Our project comprises a prolog system that will take a program P (possibly with some clauses missing or unknown) as its input loaded dynamically. The ***getAbduction/2*** predicate is run with a goal/set of goals G as its first argument and it returns in the second argument some clause(s) A which when added to the input program makes G a logical consequence of P union A. We use the clause predicate to get the body of a given goal and add the truth assignments to the list A . Upon backtracking other abducibles are derived.

We had initially proposed a stochastic abductive reasoning system but due to the semantics of probabilities, we changed this to a ***cost based abductive system*** with detection of ***minimum cost abducible.*** There are associated costs with each literal of the program which are given as facts in the input program itself. The ***findAllCost/3*** predicate is an enhancement on the getAbduction predicate. It takes a third argument C which is returned with the total cost of the abducibles A. We also have a ***findMinCost/3*** predicate that always returns the cheapest abducible among all alternates and its corresponding cost.