Constructive Logic (15-317), Spring 2022
Recitation 9: Prolog Part 2 (3-23-2022)
clogic staff

1 Warming up w/ Prolog!

In Prolog, lists are built in similarly to SML. The syntax for pattern matching on a list is `[Head | Tail]`. Using this we can implement a variety of programs for manipulating lists.

Task 1. Implement a prolog program, `mymerge/3` which merges two sorted lists.

2 Getting hotter w/ Prolog!

In Prolog, lists are built in similarly to SML. The syntax for pattern matching on a list is `[Head | Tail]`. Using this we can implement a variety of programs for manipulating lists.

Task 2. Implement a merge sorting procedure for lists, `mysort/2`, with mode (+, -) that takes in a list and merge sorts it.

Hints:

- Use a helper procedure, `split/3`, that has mode (+, -, -), that takes in a list and splits it in half into two output lists.
- There are two base cases to consider....

3 Now we're spicy w/ Prolog!

There are 4 men with last names Smith, Carpenter, Baker and Tailor. Very confusingly, their lastname does NOT correspond to their profession (either a tailor, baker, carpenter or smith). They each have a son. These sons have the same last name as their fathers, and even more confusingly, their professions do not correspond to their last names either. For example, Smith is not a smith, and SmithSon is also not a smith.

You also know:

1. No son has the same profession as his father.
2. Baker has the same profession as Carpenter's son.
3. Smith's son is a baker.

Task 3. Find the professions of the fathers and the sons using a Prolog program.

Hints:

- Use a variable for each profession you are trying to find.
- It might be useful to encode the professions in a list.
- List membership may also be useful. Recall that the following rules define list membership:

```
member(X, [X|_]).
member(X, [H|T]) :- member(X, T).
```

- You can say that A is not B.
  Example:

```
A \= B
```

Are there multiple solutions? If so, what constraints could you add to make it so there is only one solution?