Isaplanner

Property	Reason	Property	Reason
Goal30	depth(1)	Goal74	ite
Goal47	depth(1)	Goal75	incomplete
Goal52	depth(2?) + infinite loop	Goal76	similar to Goal75
Goal53	similar to Goal52	Goal79	with premise
Goal69	infinite loop + ite?	Goal80	ite + ?
Goal72	with premise		

1. `depth(i)` means that we can solve this task by setting the depth limit to `i`. These issues can be resolved by lifting the size-based heuristics to depth-level. I will try to do this latter.
2. `infinite loop` means that for some tasks, there will be loops in the egraph, making the number of possible lemmas extremely large. For example, for Goal 52, there will be an eclass like `count (S Z) (Cons Z Nil)`, `count (S Z) (Cons Z (Cons Z Nil)`, `count (S Z) (Cons Z ... (Cons Z Nil)` (all these are `Z`). It may not be a significant issue because larger lemmas will be ignored by the size-based heuristics. However, for some unknown reason, we indeed stuck into these lemmas (probably because the "discard goals" heuristics).
3. `(Goal69/75/80) ite` (not sure) we are not very good at dealing with `ite`. It seems like we have no motivation to do the propagation like `f (ite b x y) => ite b (f x) (f y)`, because we cannot construct the right term using only case-split. Besides, (I am not sure) it seems like we also cannot handle path conditions if `ite`.
4. `(Goal74) incomplete` we cannot find lemma `(len xs) = (rev (len xs))` from the original goal `(rev (drop i xs)) = (take (sub (len xs) i) (rev xs))`, as `(rev (len xs))` will never appear.
5. `with premise`, I cannot check tasks with premises because some unknown issues on my laptop.

Clam

Property	Reason	Property	Reason
Clam7	IH?	Clam34	bug?
Clam11	depth(2)	Clam50	ite
Clam14	Same as `Goal80`	Clam76	depth(1)
Clam21	incomplete	Clam78	incomplete?
Clam27	depth(2)	Clam80	incomplete?
Clam28	depth(1)	Clam81	incomplete?
Clam29	incomplete	Clam82	incomplete?
Clam31	incomplete	Clam83	incomplete
Clam32	incomplete	Clam84	cvec issue for fac
Clam33	cvec issue for fac	Clam86	cvec issue for exp

1. `IH` For `Clam7`, we need a hypothesis like `forall a, y, len (qrev xs a:ys) = S (plus (len xs) (len ys))`, but can we get this?
2. `Clam21` we should be able to prove this with depth 2, but we cannot now. Maybe there is something wrong with the size-based heuristics?
3. `Clam34` we stuck into an inifinite loop of detecting `(plus v0 (plus v1 v2)) === (plus (plus v0 v2) v1)` can discharge a goal, but making no progress in proving it.
4. `Clam29/31/78/80/81/82` are all about nestedly using `rev` and `qrev`. I am pretty sure that we have the incomplete issue on `Clam29/31` so I do not look into the later 4 tasks very carefully.