lib/basic/tableaux-tests.ath

```
##======== Some test cases for the semantic tableaux ATP ========

(load-file "tp.ath")

# (load-file "tpt.ath")

(define test1
  (if (and ?A ?B)
      (and ?B ?A)))

(define test2
  (if ?A ?A))

(define test3
  (iff (or ?A ?B)
       (or ?B ?A)))

(define test4
  (if ?A (not (not ?A))))

(define test5
  (if (if ?A ?B)
      (if (if ?B ?C)
          (if ?A ?C))))

(define test6
  (if (if ?A (if ?B ?C))
      (if ?B (if ?A ?C))))

(define test7
  (iff (if ?A (if ?B ?C))
       (if (and ?A ?B) ?C)))

(define test8
  (iff (if ?A ?B)
       (if (not ?B) (not ?A))))

(define test9
  (iff (if ?A ?B)
       (or (not ?A) ?B)))

(define test10
  (iff (not (or ?A ?B))
       (and (not ?A) (not ?B))))

(define test11
  (iff ?A
       (and ?A (or ?A ?B))))

(define test12
  (if (or (and ?A ?B)
          (and (not ?A) (not ?B)))
      (iff ?A ?B)))


(domain Nat99)

(declare (A B C D) (-> (Nat99) Boolean))

(declare (P' Q') (-> (Nat99) Boolean))
(declare (P Q) (-> (Nat99) Boolean))
(declare (R F G H) (-> (Nat99 Nat99) Boolean))


(declare E (-> (Nat99 Nat99) Boolean))


(define quant-test1
```

```
69    (forall ?x (or (A ?x)
70                   (not (A ?x)))))
71
72  (define quant-test2
73     (if (forall ?x (and (A ?x)
74                         (B ?x)))
75         (and (forall ?x (A ?x))
76              (forall ?x (B ?x)))))
77
78  (define quant-test3
79    (if (not (exists ?x (A ?x)))
80        (forall ?x (not (A ?x)))))
81
82  (define quant-test4
83    (if (and (forall ?x (A ?x))
84             (forall ?x (B ?x)))
85        (forall ?x (and (A ?x) (B ?x)))))
86
87  (define quant-test5
88    (iff (and (forall ?x (A ?x))
89              (forall ?x (B ?x)))
90         (forall ?x (and (A ?x) (B ?x)))))
91
92  (define quant-test6
93    (if (exists ?x (and (A ?x)
94                        (B ?x)))
95        (and (exists ?x (A ?x))
96             (exists ?x (B ?x)))))
97
98  (define quant-test7
99    (if (or (forall ?x (A ?x))
100           (forall ?x (B ?x)))
101       (forall ?x (or (A ?x) (B ?x)))))
102
103 (define quant-test8
104   (if (forall ?x (not (A ?x)))
105       (not (exists ?x (A ?x)))))
106
107 (define quant-test9
108   (if (not (forall ?x (A ?x)))
109       (exists ?x (not (A ?x)))))
110
111 (define quant-test10
112   (if (exists ?x (not (A ?x)))
113       (not (forall ?x (A ?x)))))
114
115 (define quant-test11
116   (if (or (exists ?x (A ?x))
117       (exists ?x (B ?x)))
118         (exists ?x (or (A ?x) (B ?x)))))
119
120 (define quant-test12
121   (if (exists ?x (or (A ?x)
122                      (B ?x)))
123       (or (exists ?x (A ?x))
124           (exists ?x (B ?x)))))
125
126 (define quant-test13
127   (iff (exists ?x (or (A ?x)
128                       (B ?x)))
129        (or (exists ?x (A ?x))
130            (exists ?x (B ?x)))))
131
132 (define quant-test14
133   (if (forall ?x (if (A ?x)
134                      (B ?x)))
135       (if (forall ?x (A ?x))
136           (forall ?x (B ?x)))))
137
138 (define quant-test15
```

```
  (if (forall ?x (if (A ?x)
                     (B ?x)))
      (if (exists ?x (A ?x))
          (exists ?x (B ?x)))))

(define quant-test16
  (if (forall ?x (if (A ?x)
                     (B ?x)))
      (if (forall ?x (A ?x))
            (exists ?x (B ?x)))))

(define quant-test17
  (if (exists ?x (forall ?y (E ?x ?y)))
      (forall ?y (exists ?x (E ?x ?y)))))


(define quant-test18
  (if (forall ?x (forall ?y (E ?x ?y)))
      (forall ?y (forall ?x (E ?x ?y)))))

(define quant-test19
  (if (exists ?x (exists ?y (E ?x ?y)))
      (exists ?y (exists ?x (E ?x ?y)))))

(define test1 (and quant-test3 (or quant-test2 quant-test8)))

(define test2 (if test1 test1))

(define test3 (and test2 test2))

(define test4 (and (if test3 test2) test3))


(define paulson1
  (if (and (or ?A ?B) (or ?A ?C))
      (or ?A (and ?B ?C))))

(define paulson2
  (iff (iff (iff ?A ?B) ?C)
       (iff ?A (iff ?B ?C))))

(define paulson3
  (if (forall ?x (E ?x ?x))
      (forall ?x (exists ?y (E ?x ?y)))))

(define ant
  (and (exists ?x (A ?x))
       (exists ?x (B ?x))))

(define cp1 (forall ?x (if (A ?x) (C ?x))))

(define cp2 (forall ?x (if (B ?x) (D ?x))))

(define cp3
  (forall ?x (forall ?y (if (and (A ?x) (B ?y))
                            (and (C ?x) (D ?y))))))

(define paulson4
  (if ant (iff (and cp1 cp2) cp3)))

(define paulson5
  (if (forall ?x (if (A ?x) (B ?x)))
      (forall ?x (if (not (B ?x)) (not (A ?x))))))

(define paulson6
  (if (and (exists ?x (A ?x))
           (forall ?x (if (A ?x) (B ?x))))
     (forall ?x (if (not (B ?x)) (not (A ?x))))))


```

```
(define pell34
  (iff (iff (exists ?x
              (forall ?y
                (iff (P ?x) (Q ?y))))
            (iff (exists ?x (Q ?x))
                 (forall ?y (P ?y))))
       (iff (exists ?x
              (forall ?y
                (iff (Q ?x) (Q ?y))))
            (iff (exists ?x (P ?x))
                 (forall ?y (P ?y))))))

(define pell35
  (exists ?x
    (exists ?y
      (if (R ?x ?y)
          (forall ?x
            (forall ?y
              (R ?x ?y)))))))

(define pell36
  (if (and (and (forall ?x
                  (exists ?y
                    (F ?x ?y)))
                (forall ?x
                  (exists ?y
                    (G ?x ?y))))
           (forall ?x
             (forall ?y
               (if (or (F ?x ?y)
                       (G ?x ?y))
                   (forall ?z
                     (if (or (F ?y ?z)
                             (G ?y ?z))
                         (H ?x ?z)))))))
      (forall ?x
        (exists ?y
          (H ?x ?y)))))


(define pell37
  (if (and (forall ?z
             (exists ?w
               (forall ?x
                 (exists ?y
                   (and (if (F ?x ?z) (F ?y ?w))
                        (and (F ?y ?z)
                             (if (F ?y ?w)
                                 (exists ?u (G ?u ?w)))))))))
           (and (forall ?x
                  (forall ?z
                    (if (not (F ?x ?z))
                        (exists ?y (G ?y ?z)))))
                (if (exists ?x
                      (exists ?y
                        (G ?x ?y)))
                    (forall ?x (H ?x ?x)))))
      (forall ?x
        (exists ?y
          (H ?x ?y)))))


(define p1
  (exists ?z
    (exists ?w
      (and (P ?z)
           (and (R ?x ?w) (R ?w ?z))))))

(define p2
  (exists ?y
```

```
279     (and (P ?y) (R ?x ?y))))
280
281 (define pell38
282   (iff (forall ?x
283          (if (and (P ?a)
284                   (if (P ?x) p2))
285              p1))
286        (forall ?x
287          (and (or (or (not (P ?a)) (P ?x)) p1)
288               (or (or (not (P ?a)) (not p2)) p1)))))
289
290 (define pell39
291   (not (exists ?x
292          (forall ?y
293            (iff (member ?y ?x)
294                 (not (member ?y ?y)))))))
295
296 (define pell40
297   (if (exists ?y
298         (forall ?x
299           (iff (member ?x ?y)
300                (member ?x ?x))))
301       (not (forall ?x
302              (exists ?y
303                (forall ?z
304                  (iff (member ?z ?y)
305                       (not (member ?z ?x)))))))))
306
307 (define pell41
308   (if (forall ?z
309         (exists ?y
310           (forall ?x
311             (iff (member ?x ?y)
312                  (and (member ?x ?z)
313                       (not (member ?x ?x)))))))
314       (not (exists ?z
315              (forall ?x
316                (member ?x ?z))))))
317
318 (define pell42
319   (not (exists ?y
320          (forall ?x
321            (iff (member ?x ?y)
322                 (not (exists ?z
323                        (and (member ?x ?z)
324                             (member ?z ?x)))))))))
325
326 (define pell43
327   (if (forall ?x
328         (forall ?y
329           (iff (= ?x ?y)
330                (forall ?z
331                  (iff (member ?z ?x)
332                       (member ?z ?y))))))
333       (forall ?x
334         (forall ?y
335           (iff (= ?x ?y)
336                (= ?y ?x))))))
337
338
339 (halt)
340
341 (!taut test1)
342 (!taut test2)
343 (!taut test3)
344 (!taut test4)
345 (!taut test5)
346 (!taut test6)
347 (!taut test7)
348 (!taut test8)
```

```
(!taut test9)
(!taut test10)
(!taut test11)
(!taut test12)

(!taut quant-test1)
(!taut quant-test2)
(!taut quant-test3)
(!taut quant-test4)
(!taut quant-test5)
(!taut quant-test6)
(!taut quant-test7)
(!taut quant-test8)
(!taut quant-test9)
(!taut quant-test10)
(!taut quant-test11)
(!taut quant-test12)
(!taut quant-test13)
(!taut quant-test14)
(!taut quant-test15)
(!taut quant-test16)
(!taut quant-test17)
(!taut quant-test18)
(!taut quant-test19)

(!taut test1)
(!taut test2)
(!taut test3)
(!taut test4)


(!taut paulson1)
(!taut paulson2)
(!taut paulson3)
(!taut paulson4)
(!taut paulson5)
(!taut paulson6)
(!taut (and paulson5 paulson6))
(!taut (and paulson4 (iff paulson5 paulson6)))
```