1. Functional test cases

1.1. Test 1: Testing basic functionality

1.1.1. Objective

This test is for testing basic ABox reasoning. It includes an ABox consistency test, some instance check tests, individuals-related tests, and basic instance-retrieval tests.

1.1.2. ABox file

`test1.abox`:

```
(related FRODO SAM friend)
(related FRODO GANDALF friend)
(instance SAM Hobbit)
(related SAM SAURON enemy)
(related SAM SARUMAN enemy)
(related GANDALF SARUMAN enemy)
(instance SARUMAN (not Orc))
(instance SAM (not Orc))
(instance GANDALF (not Orc))
(instance FRODO Creature)
(instance SAM Creature)
(instance GANDALF Creature)
(instance SAURON Creature)
(instance SARUMAN Creature)
```

1.1.3. Query file

`test1.query`:

```
(abox-consistent?)
(individual-instance? GANDALF Orc)
(individual-instance? GANDALF (not Orc))
(individual-instance? FRODO (all friend (not Orc)))
(individual-instance? FRODO (all friend Creature))
(individual-instance? FRODO (some friend Creature))
(individual-instance? FRODO (some friend (not Creature)))
(individuals-related? FRODO SARUMAN friend)
(individuals-related? FRODO GANDALF friend)
(individuals-related? FRODO GANDALF enemy)
```

```
(concept-instances (some enemy (not Orc)))
(concept-instances
 (some friend
      (and Hobbit (not Orc) (some enemy (and Orc (not Hobbit))))
 )
)
```

1.1.4. Results

Notation: T denotes a positive answer (true), while NIL denotes a non-positive answer (false or not known, where "not known" means that neither a positive, nor a negative answer may be found by the inference system). This notation is conform with the output notation of RACER. However, in the text, the cause why a NIL answer was given by the inference system (false or not known) is always specified.

- The ABox is consistent.
- The ABox contains the statement ¬Orc(GANDALF), so GANDALF is an instance of the concept Orc.
- The ABox contains the statement ¬Orc(GANDALF), so GANDALF is an instance of the concept ¬Orc.
- It is not known whether *all* friends of FRODO are instances of the concept ¬Orc, although all instances that are known friends of FRODO are not orcs.
- It is not known whether *all* friends of FRODO are instances of the concept Creature, although all instances that are known friends of FRODO are instances of the concept Creature.
- FRODO has at least one Creature friend.
- It is not known if FRODO has a friend who is not a Creature or not.
- It is not known if FRODO and SARUMAN are friends.
- FRODO and GANDALF are known to be friends.
- It is not known if FRODO and GANDALF are enemies.
- The instances who have a non-Orc enemy are GANDALF and SAM, since Saruman is not Orc, and he is an enemy of both GANDALF and SAM.

- No individuals are known to be instances of the concept in the last instance-retrieval query.

```
T
NIL
T
NIL
NIL
T
NIL
NIL
T
NIL
(GANDALF SAM)
NIL
```

1.2. Test 2: Iokaste

1.2.1. Objective

The classic ABox reasoning test presented in the Description Logics Handbook and in many other papers. The last test question is for examining if case analysis works or not. The instance retrieval query: is there an individual in the ABox who has a `Patricide` child having a non-`Patricide` child?

$$\exists hasChild.(Patricide \sqcap \exists hasChild.\neg Patricide) \qquad (1)$$

There are nine other simple cases for testing basic functionality.

1.2.2. ABox file

The ABox about the family of Iokaste:

hasChild(Iokaste,Oedipus)	hasChild(Iokaste,Polyneikes)
hasChild(Oedipus,Polyneikes)	hasChild(Polyneikes,Thersandros)
Patricide(Oedipus)	¬Patricide(Thersandros)

`test2.abox`:

```
(related IOKASTE OEDIPUS hasChild)
(related IOKASTE POLYNEIKES hasChild)
(related OEDIPUS POLYNEIKES hasChild)
(related POLYNEIKES THERSANDROS hasChild)

(instance OEDIPUS Patricide)
(instance THERSANDROS (not Patricide))
```

1.2.3. Query file

`test2.query`:

```
(abox-consistent?)
(individual-instance? IOKASTE (some hasChild Patricide))
(individual-instance? IOKASTE (some hasChild (not Patricide)))
(individual-instance? IOKASTE Patricide)
(individual-instance? IOKASTE (not Patricide))
(individual-instance? OEDIPUS Patricide)
(individual-instance? THERSANDROS (not Patricide))
(individuals-related? IOKASTE THERSANDROS hasChild)
(concept-instances (some hasChild *top*))
(concept-instances
 (some hasChild (and Patricide (some hasChild (not Patricide))))
)
```

1.2.4. Results

- The ABox is consistent.
- `IOKASTE` is known to have a `Patricide` child.
- It is not known whether `IOKASTE` has a child who is not `Patricide` (the ABox does not contain any statement involving the concept `Patricide` and individual `POLYNEIKES`).
- `IOKASTE` is not known to be an instance of the concept `Patricide`.
- `IOKASTE` is not known to be an instance of the concept ¬`Patricide`.
- `OEDIPUS` is known to be `Patricide`.
- `THERSANDROS` is known to be an instance of the concept ¬`Patricide`.
- Individuals `IOKASTE` and `THERSANDROS` are not known to be related through role `hasChild`.
- The individuals who are known to have a child are `POLYNEIKES`, `OEDIPUS` and `IOKASTE`.
- Only `IOKASTE` is known to have a `Patricide` child, who has a non-`Patricide` child. There are two cases to be examined. If `Polyneikes` is `Patricide`, he is the `Patricide` child of `IOKASTE`, who has a non-`Patricide` child, Polyneikes. Otherwise `POLYNEIKES` is not `Patricide`,

so `OEDIPUS` is the `Patricide` child of `IOKASTE`, and `OEDIPUS` has a non-`Patricide` child, `POLYNEIKES`. So in both cases `IOKASTE` is an instance of the concept in the query.

```
T
T
NIL
NIL
NIL
T
T
NIL
(POLYNEIKES OEDIPUS IOKASTE)
(IOKASTE)
```

1.3. Test 3: Case analysis involving 2 concepts

1.3.1. Objective

This test is for examining case analysis with two concepts, `A` and `B`. We will examine the instances belonging to the following instance-retrieval query concept on sevaral different ABoxes.

$$\exists R_1.(\neg A \sqcap B \sqcap \exists R_2.(A \sqcap \exists R_3.(\neg A \sqcap \neg B))) \tag{2}$$

The first ABox contains all possible arrangements of individuals `b`, `c` and `d`. For example, in the first line of the table, `a` and `b` are related through role R_1, `b` and `c` are related through role R_2 and instances `c` and `d` are related through role R_3. According to the specified ABox, nothing more is known about these individuals. The objective of this case is to show that it is not enough to enumerate all the different arrangements of three individuals to get a positive answer.

$\top$	¬A, B	A	¬A, ¬B
a	b	c	d
a	b	d	c
a	c	b	d
a	c	d	b
a	d	b	c
a	d	c	b

The second ABox examines one individual who is not known to be an instance of `A, ¬A, B` or `¬B`. All the other instances are known to be instances of the concepts present in the heading cell of the table where the individual is present. When making a case analysis on concepts A and B for individual c, the cases (A ⊓*B*)(c), (¬A ⊓ B)(c), (A ⊓ ¬B)(c) and (¬A ⊓ ¬B)(c) have to be examined. In this test, the cases (A ⊓ ¬B)(c) and (A ⊓ B)(c) are merged into one case A(c).

⊤	¬A, B	A	¬A, ¬B	Assertions
a	x1	y1	z1	¬A(x1) B(x1) A(y1) ¬A(z1) ¬B(z1)
b	*c*	y2	z2	A(y2) ¬A(z2) ¬B(z2)
b	x3	*c*	z3	¬A(x3) B(x3) ¬A(z3) ¬B(z3)
b	x4	y4	*c*	¬A(x4) B(x4) A(y4)

The third ABox demonstrates that it is not enough to examine the asserted statements A(c) and (¬A ⊓ B)(c) for the individual c when performing a case analysis. If we make a case analysis on individual c and concepts A and B, we need to cover all four possibilities A ⊓ B, A ⊓ ¬B, ¬A ⊓ B and ¬A ⊓ ¬B. In this test the branch ¬A ⊓ ¬B is not covered.

⊤	¬A, B	A	¬A, ¬B	Assertions
a	*c*	y	z	A(y) ¬A(z) ¬B(z)
a	x	*c*	z	¬A(x) B(x) ¬A(z) ¬B(z)

The fourth and fifth ABoxes test a case analysis for two individuals, c and d, where the ABox does not contain any concept-instance assertions for instance c, and contains only the concept-instance assertion ¬A(d) for instance d.

⊤	¬A, B	A	¬A, ¬B	Assertions
a	*c*	y1	*d*	A(y1) ¬A(d)
a	*d*	y2	*c*	¬A(d) A(y2)
a	x3	*c*	z3	A(x3) ¬B(x3) ¬A(z3) ¬B(z3)

⊤	¬A, B	A	¬A, ¬B	Assertions
a	*c*	y1	*d*	A(y1) ¬A(d)
a	x2	*c*	z2	¬A(x2) B(x2) ¬A(z2) ¬B(z2)
a	x3	y3	*c*	A(x3) ¬B(x3) A(y3)
a	*d*	y4	z4	A(y4) ¬A(z4) ¬B(z4)

1.3.2. ABox files

- `test3a.abox`:

```
(related a b R1)
```

```
(related a c R1)
(related a d R1)
(related b c R2)
(related b d R2)
(related c b R2)
(related c d R2)
(related d b R2)
(related d c R2)
(related c d R3)
(related d c R3)
(related b d R3)
(related d b R3)
(related b c R3)
(related c b R3)
```

- test3b.abox:

```
(related a  x1 R1)
(related b  c  R1)
(related b  x3 R1)
(related b  x4 R1)

(instance x1 (not A))
(instance x1 B)
(instance x3 (not A))
(instance x3 B)
(instance x4 (not A))
(instance x4 B)

(related x1 y1 R2)
(related c  y2 R2)
(related x3 c  R2)
(related x4 y4 R2)

(instance y1 A)
(instance y2 A)
(instance y3 A)

(related y1 z1 R3)
(related y2 z2 R3)
(related c  z3 R3)
```

```
(related y4 c  R3)

(instance z1 (not A))
(instance z1 (not B))
(instance z2 (not A))
(instance z2 (not B))
(instance z3 (not A))
(instance z3 (not B))
```

- `test3c.abox`:

```
(related a c R1)
(related c y R2)
(related y z R3)
(related a x R1)
(related x c R2)
(related c z R3)

(instance x (not A))
(instance x B)
(instance y A)
(instance z (not A))
(instance z (not B))
```

- `test3d.abox`:

```
(related a c R1)
(related c y1 R2)
(related y1 d R3)
(related a d R1)
(related d y2 R2)
(related y2 c R3)
(related a x3 R1)
(related x3 c R2)
(related c z3 R3)

(instance d (not A))
(instance x3 (not A))
(instance x3 B)
(instance y1 A)
(instance y2 A)
```

```
(instance z3 (not A))
(instance z3 (not B))
```

- `test3e.abox`:

```
(related a  c  R1)
(related c  y1 R2)
(related y1 d  R3)

(related a  x2 R1)
(related x2 c  R2)
(related c  z2 R3)
(related a  x3 R1)
(related x3 y3 R2)
(related y3 c  R3)

(related a  d  R1)
(related d  y4 R2)
(related y4 z4 R3)

(instance d (not A))

(instance x2 (not A))
(instance x2 B)
(instance x3 (not A))
(instance x3 B)

(instance y1 A)
(instance y3 A)
(instance y4 A)

(instance z2 (not A))
(instance z2 (not B))
(instance z4 (not A))
(instance z4 (not B))
```

1.3.3. Query file

The test questions are the same for all the ABoxes:
`test3.query`:

```
(abox-consistent?)
(concept-instances
 (some R1
       (and (not A)
            B
            (some R2 (and A (some R3 (and (not A) (not B)))))
       )
 )
)
```

1.3.4. Results

All ABoxes are consistent, so the answer returned for the first question is always `T` for all ABoxes.

- `test3a.abox`: the answer for the instance-retrieval question is obviously `NIL`, because `a` may not be an instance of the query-concept if nothing is known about individuals `b`, `c` and `d`.

  ```
  T
  NIL
  ```

- `test3b.abox`: instances `a` and `b` are both instances of the query-concept: `a` is found to be an instance even in closed world reasoning, while finding `b` as an answer requires case analysis.

  ```
  T
  (A B)
  ```

- `test3c.abox`: Since there are not enough knowledge to make a case analysis on concept `c`, the result is expected to be `NIL`.

  ```
  T
  NIL
  ```

- `test3d.abox`: If we made a case analysis only on concept `c` or `d` and all other information about the other concepts were known, the case analysis would be succesful in both cases and `a` would be returned as an answer. But here a case analysis is needed for both concepts, so there are not enough assertions in the ABox to find `a` as an answer.

```
T
NIL
```

- `test3e.abox`: A successful case analysis on concepts `A` and `B` for individuals `c` and `d`.

```
T
(A)
```

1.4. Test 4: Case analysis involving two individuals

Let us suppose that we have a concept `A`, and two individuals, `c` and `d`. According to the ABox, these two instances are not known to be instances of concepts `A` or `¬A`. We are looking for the answer of the following query:

$$\exists R1.(A \sqcap \exists R2.(\neg A \sqcap \exists R3.(A \sqcap \exists R4.\neg A))). \tag{3}$$

There are other instances in the ABox, which are known to be instances of either concept `A` or concept `¬A`.

$\top$	¬A	A	¬A	A	Assertions
a	x1	*c*	z1	*d*	¬A(x1), ¬A(z1)
a	x2	*c*	*d*	w1	¬A(x2), A(w1)
a	*c*	y1	z2	*d*	A(y1), ¬A(z2)
a	*c*	y2	*d*	w2	A(y2), A(w2)
b	x3	y3	z3	w3	¬A(x3), A(y3), ¬A(z3), A(w3)
x	x4	c	z4	w4	¬A(x4), ¬A(z4), A(w4)
x	c	y4	z4	w4	A(y4), ¬A(z4), A(w4)

1.4.1. ABox file

`test4.abox`:

```
(related a x1 R1)
(related a x2 R1)
(related b x3 R1)
(related x x4 R1)
(related a c  R1)
(related x c  R1)

(related x1 c  R2)
```

```
(related x2 c  R2)
(related c  y1 R2)
(related c  y2 R2)
(related x3 y3 R2)
(related x4 c  R2)
(related c  y4 R2)

(related c  z1 R3)
(related c  d  R3)
(related y1 z2 R3)
(related y2 z  R3)
(related y3 z3 R3)
(related c  z4 R3)
(related y4 z4 R3)

(related z1 d  R4)
(related d  w1 R4)
(related z2 d  R4)
(related d  w2 R4)
(related z3 w3 R4)
(related z4 w4 R4)

(instance x1 (not A))
(instance x2 (not A))
(instance x3 (not A))
(instance x4 (not A))
(instance z1 (not A))
(instance z2 (not A))
(instance z3 (not A))
(instance z4 (not A))

(instance y1 A)
(instance y2 A)
(instance y3 A)
(instance y4 A)
(instance w1 A)
(instance w2 A)
(instance w3 A)
(instance w4 A)
```

1.4.2. Query file

`test4.query`:

```
(abox-consistent?)
(concept-instances
 (some R1
  (and (not A)
       (some R2 (and A (some R3 (and (not A) (some R4 A)))))
  )
 )
)
```

1.4.3. Results

The ABox is consistent.

For the second query, `b` is found to be an instance of the query-concept without case analysis, `x` is found to be an instance through a case-analysis on individual `c`, and `a` is found to be an instance through a case-analysis on instances `c` and `d`.

```
T
(X A B)
```

1.5. Test 5: Exhaustive case analysis involving two concepts and two individuals

1.5.1. Objective

The test cases of test 3 and 4 are further refined to yield a case for testing case analysis for two individuals c and d, where both individuals occur in each row of the query-table. Furthermore, there are two concepts in the test, `A` and `B`, for which the ABox does not contain of the assertions `A(c)`, `B(c)`, `A(d)`, `B(d)`, ¬`A(c)`, ¬`B(c)`, ¬`A(d)` or ¬`B(d)`.

For every individual having a name that consists of two characters, two assertions are specified in the ABox: each individual is an instance of the two concepts present in the heading of the table, in the column where the individual is present. For instance, for the individual `l6`, the ABox contains the assertive knowledge `A(l6)` and `B(l6)`.

Instance-retrieval query: $\exists R_1.(\neg A \sqcap \neg B \sqcap \exists R_2.(\neg A \sqcap B \sqcap \exists R_3.(A \sqcap \neg B \sqcap \exists R_4.(A \sqcap B \sqcap \exists R_5.(\neg A \sqcap \neg B \sqcap \exists R_6.(\neg A \sqcap B \sqcap \exists R_7.(A \sqcap \neg B \sqcap \exists R_8.(A \sqcap B))))))))$

⊤	¬A, ¬B	¬A, B	A, ¬B	A, B	¬A, ¬B	¬A, B	A, ¬B	A, B
a	i0	j0	k0	c	m0	n0	o0	d
a	i1	j1	k1	c	m1	n1	d	p1
a	i2	j2	k2	c	m2	d	o2	p2
a	i3	j3	k3	c	d	n3	o3	p3
a	i4	j4	c	l4	m4	n4	o4	d
a	i5	j5	c	l5	m5	n5	d	p5
a	i6	j6	c	l6	m6	d	o6	p6
a	i7	j7	c	l7	d	n7	o7	p7
a	i8	c	k8	l8	m8	n8	o8	d
a	i9	c	k9	l9	m9	n9	d	p9
a	ia	c	ka	la	ma	d	oa	pa
a	ib	c	kb	lb	d	nb	ob	pb
a	c	jc	kc	lc	mc	nc	oc	d
a	c	jd	kd	ld	md	nd	d	pd
a	c	je	ke	le	me	d	oe	pe
a	c	jf	kf	lf	d	nf	of	pf

1.5.2. ABox file

See `test5.abox` attached to this document.

1.5.3. Query file

`test5.query`

```
(abox-consistent?)
(concept-instances
 (some R1
  (and (not A) (not B)
   (some R2
    (and (not A) B
     (some R3
      (and A (not B)
       (some R4
        (and A B
         (some R5
          (and (not A) (not B)
           (some R6
            (and (not A) B
             (some R7
```

```
              (and A (not B)
               (some R8 (and A B)
               )
              )
             )
            )
           )
          )
         )
        )
       )
      )
     )
    )
   )
  )
 )
)
```

1.5.4. Results

The ABox is consistent, so the answer for the first question is `T`.

For the second query, instance `a` is a solution, because whatever we suppose for individuals `c` and `d`, a chain of R_i-successors always exists that satisfy the concept in the instance-retrieval query. So `a` is found to be a solution through an analysis of 16 cases.

```
T
(A)
```