
1. At page 61, add just before the header of section 3.1: “It is well known that the RDFS closure is infinite, and we will describe how we ignore some of the RDFS rules which produce either trivial results or ignore derivations that are considered bad style. For similar reasons, we do not add the RDFS axiomatic triples to the input data.”

2. At page 64, the third antecedent of rule 14a of Table 3 is “u p w”.

3. At page 64, the head of the rule 14b of Table 3 is “u p w”.

4. At page 67, add at the end of section 6.1: “In the latest version of the code, the execution of OWL reasoning requires one more job to finish because of an implementation bug of the incremental reasoning procedure”.

5. At page 68, the header of the third column should be “Throughput (Ktps)”

6. At page 72, add at the end of the first paragraph of appendix A.1: “As usual, our pseudocode omits details that are not essential for human understanding of the algorithm, such as variable declarations, datatypes and some subroutines (http://en.wikipedia.org/wiki/Pseudocode)”

7. At page 71, first line of Section 8.2: should be “RDFS/OWL Horst” instead of “OWL Horst”

8. In Appendix A, Algorithm 3 (page 72) should read:

```plaintext
rdfs_reasoning(data) {
    derived = apply_job(data, SUBPROP);
    derived += apply_job(data + derived, DOMAINRANGE);
    derived = clean_duplicates(data, derived);
    derived += apply_job(data + derived, SUBCLASS);
    if (derived_special_cases_no_loop(derived) == true)
        derived += apply_job(data + derived, SPECIAL_CASES);
    if (derived_special_cases_with_loop(derived) == true)
        derived += rdfs_reasoning(data + derived);
    return derived
}
```

9. In Appendix A, Algorithm 6 (page 73) lines 3 and 6 should output value.subject instead of value.predicate

10. In Appendix A (page 72), add to the paragraph on SUBCLASS: “The pseudocode of Algorithm 6 does not mention the computation of rules 12 and 13, because their execution is trivial.”

11. In Appendix A, Algorithm 10 (page 74) should be:

```plaintext
map(key, edge):
    emit(edge.from, edge.to);
    emit(edge.to, edge.from);
reduce(key, values):
    toNodes.empty(); // edges to other nodes
    foundReplacement = false
    for (value in values)
        if (value < key)
            if (foundReplacement)
                toNodes.add(key);
            foundReplacement = true;
            key = value;
        else if (value > key) toNodes.add(value);
    for (to in toNodes)
        emit(null, {key, to});
```

Points 2, 3, 4, 5 and 11 are bugs not mentioned in the letter, but discovered by us some time ago.