Distributed Reasoning with Conflicts in a Peer-to-Peer Setting

January 31, 2008

1 Definitions

We assume a peer-to-peer system P as a collection of peer local theories:

$$P = \{P_i\}, i = 1, 2, ..., n$$

Each peer has a proper distinct vocabulary V_{P_i} and a unique identifier i. Each local theory is a set of rules that contain only local literals (literals from the local vocabulary). These rules are of the form:

$$r_i : a_i^1, a_i^2, ...a_i^{n-1} \rightarrow a_i^n$$

where i denotes the peer identifier.

Each peer also defines mappings that associate literals from its own vocabulary (*local literals*) with literals from the vocabulary of other peers (*remote literals*). The acquaintances of peer P_i, $ACQ(P_i)$ are the set of peers that at least one of P_i's mappings involves at least one of their local literals. The mappings are rules of the form:

$$m_i : a_i^1, a_j^2, ...a_k^{n-1} \Rightarrow a^n$$

The above mapping rule is defined by P_i, and associates some of its own local literals with some of the literals defined by P_j, P_k and other system nodes. Literal a^n may belong to whichever of these system nodes' vocabulary.

Finally, each node P_i defines a trust level order T_i, which includes a subset of the system nodes, and expresses the trust that P_i has on the other system nodes. This is of the form:

$$T_i = [P_k, P_l, ..., P_n]$$

The nodes that are not included in T_i are less trusted by P_i than those that are part of this ordering list.

2 Problem Statement

Given a peer-to-peer system P, and a query about literal x_i issued at peer P_i, find the truth value of x_i considering P_i's local theory, its mappings and the theories of the other nodes in the system.

We assume that the local theories are consistent, but this is not necessarily true for the case of the unified theory $T(P)$, which is the collection of the theories (local rules and mappings) of the system nodes. The inconsistencies result from interactions between local theories and are caused by mappings.

An example of such conflicts derives in the following system of theories, which are defined by peers P_1, P_2 and P_3:

P_1
$r_{11} : a_1 \rightarrow x_1$
$m_{11} : a_2 \rightarrow a_1$
$m_{12} : a_3 \rightarrow \neg a_1$

P_2
$r_{21} :\rightarrow a_2$

P_3
$r_{31} :\rightarrow a_3$

P_i's theory is locally consistent, but with the addition of the the two mapping rules (m_{11}, m_{12}), which associate the literals of P_1,P_2 and P_3, a conflict about literal a_1 derives from the interaction of the three theories.

3 The 1st Approach

3.1 The Algorithm

The algorithm that we propose follows four main steps. In the first step, it uses P_i's local theory to determine the truth value of x_i. If x_i or its negation, $\neg x_i$ derive from the peer's local theory, the algorithm terminates returning Yes/No respectively, without considering the peer's mappings.

In the second step, if neither x_i nor $\neg x_i$ derive from the local theory, the algorithm also uses P_i's mappings. It collects all the rules that support x_i. For each such rule, it checks the truth value of the literals in its body. For each local/remote literal, it issues similar queries (recursive calls of the algorithm) to P_i (local literals) or to the appropriate P_i's acquaintances (remote literals). To avoid circles, before each new call, the algorithm checks if the same query has been issued before, during the algorithm call. At the end of this step, the algorithm builds the mapping supportive set of x_i; this contains the set of *foreign literals* (literals that are defined by peers that belong in $ACQ(P_i)$) that are contained in the body of the P_i's mapping rules, which can be applied to prove x_i in the absence of any possible contradictions.

The third step involves the rules that contradict x_i. The algorithm builds the mapping conflicting set of x_i, by collecting the foreign literals in the bodies of the mapping rules that are used to support $\neg x_i$.

Finally, the algorithm determines the truth value of x_i by comparing the supportive and conflicting sets. To compare two mapping sets, there are several different approaches. One approach is to compare the number of distinct peers, n_p, that at least one of their local literals is contained in the mapping set. The mapping set with the smallest n_p is considered to be stronger. Another general approach is to have each peer define an ordering between the peers of the system based on the trust it has on them. According to this ordering, a literal a_k is considered to be stronger than b_l from P_i's viewpoint if P_i trusts P_k more than P_l. Having defined this ordering, we just need a function that calculates the strength of a mapping set based on the strength of the literals it contains. In the following algorithm, the strength of a mapping set is determined by the weakest literal in this set. In the followings, we denote as:

r_i^l: a local rule of P_i

r_i^m: a mapping rule of P_i

r_i^{lm}: a rule (local/mapping) of P_i

R_m: the set of all mapping rules

$R_s(x_i)$: the set of supportive rules for x_i

$R_c(x_i)$: the set of conflicting rules for x_i

When a node P_i receives a query about x_i, it runs the **P2P_DR** algorithm. The algorithm parameters are:

x_i: the queried literal

P_0: the peer that issued the query

P_i: the local node

SS_{x_i}: the set of supportive foreign literals for x_i (initially empty)

CS_{x_i}: the set of conflicting foreign literals for x_i (initially empty)

$Hist_{x_i}$: the list of pending queries of the form: $[x_1, ..., x_i]$

Ans_{x_i}: the answer returned for x_i (initially empty)

T_i: the trust level order of P_i

P2P_DR$(x_i, P_0, P_i, SS_{x_i}, CS_{x_i}, Hist_{x_i}, Ans_{x_i}, T_i)$

if $\exists r_i^l \in R_s(x_i)$ **then**
    $localHist_{x_i} \leftarrow [x_i]$
    run $local\_alg(x_i, localHist_{x_i}, localAns_{x_i})$
    if $localAns_{x_i} = Yes$ **then**
        $Ans_{x_i} \leftarrow localAns_{x_i}$
        terminate
    end if
end if
if $\exists r_i^l \in R_c(x_i)$ **then**
    $localHist_{x_i} \leftarrow [x_i]$
    run $local\_alg(\neg x_i, localHist_{x_i}, localAns_{\neg x_i})$
    if $localAns_{\neg x_i} = Yes$ **then**
        $Ans_{x_i} \leftarrow \neg localAns_{\neg x_i}$
        terminate

```
    end if
end if
for all r_i^{lm} ∈ R_s(x_i) do
    SS_{r_i} ← {}
    for all b_t ∈ body(r_i^{lm}) do
        if b_t ∈ Hist_{x_i} then
            stop and check the next rule
        else
            Hist_{b_t} ← Hist_{x_i} ∪ b_t
            run P2P_DR(b_t, P_i, P_t, SS_{b_t}, CS_{b_t}, Hist_{b_t}, Ans_{b_t}, T_t)
            if Ans_{b_t} = No then
                stop and check the next rule
            else if Ans_{b_t} = Yes and b_t ∉ V_i then
                SS_{r_i} ← SS_{r_i} ∪ b_t
            else
                SS_{r_i} ← SS_{r_i} ∪ SS_{b_t}
            end if
        end if
    end for
    if SS_{x_i} = {} or Stronger(SS_{r_i}, SS_{x_i}, T_i) = SS_{r_i} then
        SS_{x_i} ← SS_{r_i}
    end if
end for
if SS_{x_i} = {} then
    return Ans_{x_i} = No and terminate
end if
for all r_i^{lm} ∈ R_c(x_i) do
    SS_{r_i} ← {}
    for all b_t ∈ body(r_i^{lm}) do
        if b_t ∈ Hist_{x_i} then
            stop and check the next rule
        else
            Hist_{b_t} ← Hist_{x_i} ∪ b_t
            run P2P_DR(b_t, P_i, P_t, SS_{b_t}, CS_{b_t}, Hist_{b_t}, Ans_{b_t}, T_t)
            if Ans_{b_t} = No then
                stop and check the next rule
            else if Ans_{b_t} = Yes and b_t ∉ V_i then
                SS_{r_i} ← SS_{r_i} ∪ b_t
            else
                SS_{r_i} ← SS_{r_i} ∪ SS_{b_t}
```

end if
end if
end for
if $CS_{x_i} = \{\}$ *or* $Stronger(SS_{r_i}, CS_{x_i}, T_i) = SS_{r_i}$ **then**
$CS_{x_i} \leftarrow SS_{r_i}$
end if
end for
if $CS_{x_i} = \{\}$ **then**
return $Ans_{x_i} = Yes$ and SS_{x_i} and terminate
end if
if $Stronger(SS_{x_i}, CS_{x_i}, T_i) = SS_{x_i}$ **then**
return $Ans_{x_i} = Yes$ and SS_{x_i}
else
return $Ans_{x_i} = No$
end if

$local\_alg(x_i, localHist_{x_i}, localAns_{x_i})$ is used to determine if x_i is a consequence of P_i's local theory. The algorithm parameters are:

x_i: the queried literal

$localHist_{x_i}$: the list of pending queries in P_i of the form: $[x_i^1, ..., x_i^m]$

$localAns_{x_i}$: the local answer returned for x_i (initially No)

local_alg$(x_i, localHist_{x_i}, localAns_{x_i})$
for all $r_i^l \in R_s(x_i)$ **do**
if $body(r_i^l) = \{\}$ **then**
return $localAns_{x_i} = Yes$
terminate
else
for all $b_i \in body(r_i^l)$ **do**
if $b_i \in localHist_{x_i}$ **then**
stop and check the next rule
else
$localHist_{b_i} \leftarrow localHist_{x_i} \cup b_i$
run $local\_alg(b_i, localHist_{b_i}, localAns_{b_i})$
end if
end for
if for every b_i: $localAns_{b_i} = Yes$ **then**
$localAns_{x_i} \leftarrow Yes$

terminate
end if
end if
end for

The $Stronger(S, C, T)$ function is used by a peer P to check which of S,C sets of mappings is *stronger*, based on T.

Stronger(S, C, T)

$a^w \leftarrow a_k \in S$ s.t. *for all* $a_i \in S : P_k$ does not precede P_i in T)
$b^w \leftarrow a_l \in C$ s.t. *for all* $b_j \in C : P_l$ does not precede P_j in T)
if P_k precedes P_l in T **then**
    $Stronger = S$
else if P_l precedes P_k in T **then**
    $Stronger = C$
else
    $Stronger = None$
end if

3.2 Algorithm Properties

3.2.1 Termination

Theorem 1 *The P2P_DR algorithm always terminates.*

Proof. We assume that there are a finite number of nodes in the system, each of which with a finite number of literals in its vocabulary. As a consequence, there are a finite number of (local or mapping) rules that a peer can define. At each recursive call of the algorithm, we augment the history with a new pending query q_i, where q_i is one of P_i's local literals, and P_i is one of the system nodes. Each call of the algorithm terminates either by computing and returning a Yes/No answer (based on the provability of q_i) or by detecting a cycle. If the algorithm did not terminate, it would have to make indefinite recursive calls, adding each time a new query to the history, without ever returning an answer or detecting a cycle. However, this is impossible, because: (a) the number of recursive calls is bounded by the total finite number of literals in the system; and (b), there can be a finite number of independent (with different history) algorithm calls that the system may process. These are bounded by the total finite number of rules in the system. Consequently, the algorithm will eventually terminate.

3.2.2 Algorithm Optimizations

To reduce the complexity of the basic algorithm with regard to the number of messages that the system nodes have to exchange, and the computational overhead of the algorithm on each system node, we can optimize the algorithm as follows:

Each node is required to retain two states: (a)the state of queries it has been requested to process, INC_Q; this contains tuples of the form $(q_i, Ans_{q_i}, localAns_{q_i})$, where q_i is the queried local literal, and Ans_{q_i} and $localAns_{q_i}$ are $true/false$ in the case the node has completed the computation, or *undetermined* otherwise; and (b) the state of queries it has issued to other peers, OUT_Q (of the same form). Before sending a query to one of $P_j \in ACQ(P_i)$, P_i checks if the same query is in its OUT_Q. If this is the case, it retrieves the answer stored in OUT_Q in case the answer has a $true/false$ value; otherwise it suspends until the pending query returns a $true/false$ answer. When a new query is issued at P_i, the node checks if the same query is in its INC_Q. If it is, the node returns the stored $true/false$ answer for that query if this has already been computed, or suspends the new query until the pending query returns a $true/false$ answer. The space overhead of INC_Q is proportional to the number of local literals in P_i, while the size of OUT_Q is in the worst case proportional to the number of peers $P_j \in ACQ(P_i)$ and the number of their local literals ($a_j \in V_j$). The two states need to be updated every time a new query is issued at the system from an external source (we assume that the state of the network remains unchanged during the computation of each such query).

In order to reduce the overhead of searching in $Hist$ and OUT_Q, these can be structured as collections of fields, where each field corresponds to a peer identifier. In this way, checking whether q_i is included in $Hist$ (or whether $(q_i, answer)$ is in OUT_Q) requires checking only the $i - field$ of $Hist$ (or of OUT_Q).

3.2.3 Number of Messages

Theorem 2 *The total number of messages that need to be exchanged between the system nodes for the computation of a single query with regard to the total number of system nodes is in the worst case* $O(n^2)$.

Proof. With the optimizations that we describe in the previous section, each node will have to make at most one query for each of the remote literals that appear in the body of its mapping rules. In the case, that a peer P_i needs to apply all mapping rules during a query evaluation process, P_i will have to make $O(n_{ACQi} \times n_l)$ queries, where n_{ACQi} is the number of $P_j \in ACQ(P_i)$ and n_l is the maximum number of literals that each of these nodes may define. So, the total number of messages that need to be exchanged for the computation of a single query is $O(n \times n_{ACQ} \times n_l)$, where n_{ACQ} is the maximum number of acquaintances a system node may have. In the worst case, that all peers have defined mappings that involve all the other system nodes, the total number of messages is $O(n \times n \times n_l) = O(n^2)$ (assuming that the number of nodes is the most critical parameter in the system).

3.2.4 Single Node Complexity

In this section, we estimate the computational complexity of the distributed algorithm on a single node.

The first part of the algorithm requires checking the local rules of a peer, to determine if the query can be locally resolved. In the worst case, to reach to an answer, the algorithm will have to use all its local rules. For each literal in the body of each such rule, the algorithm (a) checks if a query about it is contained in $localHist$, (b) checks if it is included in INC_Q; and (c) if not, it issues a recursive call of the algorithm to compute a local answer. Considering the structural form of $Hist$ and INC_Q, each of the (a) and (b) steps require $O(n_l)$ checks (matching operations), where n_l is the maximum number of literals a node may define. Thus, the total matching operations required for each rule are $O(n_l^{rloc} \times n_l)$, where n_l^{rloc} is the number of literals in the body of a local rule, and the total computational complexity of the local answer resolution is in the worst case $O(n_{rloc} \times n_l^{rloc} \times n_l)$, where n_{rloc} is the maximum number of local rules a peer may define.

With the optimizations described in Section 4.2.1, in the worst case, each peer will have to compute the truth value of all its local literals at most once. This means that it will have to build and compare the Supportive Set and Conflicting Set for each of its local literals.

To build the Supportive Set of a literal x_i (SS_{x_i}), a peer has to compute the Supportive Sets of the rules that support it (SS_{r_i}). For each literal in the body of each such rule, the algorithm (a) checks if a query about it is contained in $Hist$, (b) checks if it is included in INC_Q or OUT_Q; and (c) if not, it issues a recursive call of the algorithm to compute an answer about its truth value. Considering the structural form of $Hist$ and OUT_Q, each of (a) and (b) steps require at most $O(n_l^r \times n_l)$ operations for each rule, where n_l^r is the number of literals in the body of a local / mapping rule. Building the Supportive Set of a rule then requires only unifying the Supportive Sets of its body literals. Given that each Supportive Set may contain (in the worst case) $O(n_{ACQ} \times n_l)$ literals, where n_{ACQ} is the maximum number of acquaintances a system node may have, the complexity of building this set is $O(n_l^r \times n_{ACQ} \times n_l)$.

Computing the Supportive Set of a literal (SS_{x_i}), given the Supportive Sets of its supportive rules (SS_{r_i}), requires finding the *strongest* SS_{r_i} through the *Stronger* function. Considering that (a) the complexity of this function is proportional to the total number of elements of its two set arguments; and (b) each Supportive Set may contain (in the worst case) $O(n_{ACQ} \times n_l)$ literals,comparing two SS_{r_i} has a $O(n_{ACQ} \times n_l)$ overhead.

Thus, the overall complexity of building the supportive set of a literal is $O(n_{r_{xi}} \times (n_l^r \times n_l + n_l^r \times n_{ACQ} \times n_l + n_{ACQ} \times n_l)) = O(n_{r_{xi}} \times n_l^r \times n_{ACQ} \times n_l)$, where $n_{r_{xi}}$ is the number of rules that support it. This is also equal to the complexity of computing the conflicting set of the same literal (CS_{x_i}). The complexity of comparing the two sets through the *Stronger* function to determine about the truth value of x_i is $O(n_{ACQ} \times n_l)$, considering that each such set may contain $O(n_{ACQ} \times n_l)$ literals.

During the execution of the algorithm, a peer may have to compute the supportive and conflicting sets, and the truth value of all its local literals. Putting it all together, the overall complexity of the algorithm on a single node is

$$O(n_{rloc} \times n_l^{rloc} \times n_l + n_r \times n_l^r \times n_{ACQ} \times n_l + n_l \times n_{ACQ} \times n_l)$$

n_{rloc} is the number of local rules defined by a peer
n_r is the number of (local and mapping) rules defined by a peer
n_l^r is the number of literals in the body of a rule
n_l^{rloc} is the number of literals in the body of a local rule
n_l is the number of literals defined by one peer
n_{ACQ} is the number of a peer's acquaintances

Assuming that

(a) $n_l^r = O(n_{ACQ} \times n_l)$ (the body of a rule may contain all the literals defined in the local theory or in the theory of the peer's acquaintances); and

(b) $n_l^{rloc} = O(n_l)$ (the body of a local rule may involve all local literals), the overall complexity is

$$O(n_{ACQ}^2 \times n_l^2 \times n_r)$$

In the worst case, that that all peers have defined mappings that involve all the other system nodes: $n_{ACQ} = O(n)$, and the overall complexity is

$$O(n^2 \times n_l^2 \times n_r)$$

3.3 Equivalent Unified Defeasible Theory

The next issue of this study is to build a unified defeasible theory $T(P)$, which is equivalent to the distributed theories with regard to the conclusions that they draw. A naive approach would be to just copy the local and mapping rules of each node in a single theory, and represent the local rules as strict rules, and the mappings as defeasible rules of a defeasible theory. This approach suffers from the following problems:

If a query about a literal x_i, which is part of P_i's local theory is issued to a different peer, P_j, the distributed algorithm will return No as a result. In the case of the unified theory, we could have a different result based on the rules that derive from P_i. We can deal with this problem by employing an additional routing mechanism, which is responsible for routing queries to the appropriate peers. Given the fact that each peer defines its own vocabulary, the routing mechanism is able to figure out the peer that has defined each queried literal, just by reading the name of the literal. If the query is issued to the appropriate node, then the same set of (local) rules will be considered in the first step of the algorithm in both cases.

If the algorithm cannot return an answer based on P_i's local rules, it will use P_i's mappings. For a query about literal x_i, the algorithm will consider the supportive and conflicting rules for x_i, which are defined in P_i. However, other nodes in the system may also have defined mapping rules that support/contradict x_i. These rules will not be considered by the distributed algorithm, but are part of the unified theory. To achieve consistency, we have to remove all mapping rules that support or contradict remote literals (literals that are defined in a different theory from that which defines the mapping rule).

If there is a conflict between two or more rules that support contradictory conclusions (say x_i and $\neg x_i$), the distributed algorithm collects the supportive and conflicting sets of foreign literals, and decides based on the strength of these sets. In the unified theory, we must find a way to model these strengths (levels of trust) as priorities between the conflicting rules.

Considering these problems, we build the unified defeasible theory $T_v(P)$ as follows:

1. The local rules of each peer's theory are represented as strict rules in $T_v(P)$.
2. The mapping rules of each peer are represented as defeasible rules in $T_v(P)$.
3. We remove from $T_v(P)$ all mapping rules that support or contradict remote literals. We do that, by comparing the name of the rule (which includes the id of the peer that has defined the rule), with the name of the literal at the head of the rule (which includes the id of the peer that has defined the literal).
4. We add priorities between the conflicting rules according to the derivation process described below.

Priorities

The derivation of priorities between conflicting rules in $T_v(P)$ is a finite sequence $Pr = (Pr(1), ..., Pr(n))$, where each $Pr(i)$ can be one of the followings:

- The supportive set of a rule in $T_v(P)$ (a set of literals).
- A priority relation between two conflicting rules in $T_v(P)$
- The supportive set of a literal in $T_v(P)$ (a set of literals).

Assuming that the first i steps of this derivation have computed $Pr(1...i)$, which is the initial part of the sequence Pr of length i, the next part of this sequence ($Pr(i+1)$) will be either the supportive set of a rule (S_{r_i}), or a priority relation ($r_i > s_i$), or the supportive set of a literal (S_{a_i}). In the process that we describe below, w can be thought as an element, which is weaker than any literal of the context theories. We use this element to build sets of literals that cannot be *stronger* than any other set.

If $Pr(i+1) = S_{r_i}$ *then either*
(α) $S_{r_i} = (\bigcup S_{a_i}) \cup (\bigcup a_j)$, *and*
$\forall a_i$: $a_i \in V_i$, $a_i \in body(r_i)$, $S_{a_i} \in Pr(1...i)$ *and*
$\forall a_j$: $a_j \notin V_i$, $a_j \in body(r_i)$, $S_{a_j} \in Pr(1...i)$, $w \notin S_{a_j}$ *or*
(β) $S_{r_i} = \{w\}$, *and*
$\exists a_j$, s.t. $a_j \notin V_i$, $a_j \in body(r_i)$, $S_{a_j} \in Pr(1...i)$, $w \in S_{a_j}$

If $Pr(i+1) = r_i > s_i$ *then*
$S_{r_i}, S_{s_i} \in Pr(1...i)$ *and* r_i, s_i *are conflicting* ($r_i \in R[a_i] \Leftrightarrow s_i \in R[\neg a_i]$) *and*
$S_{r_i}, S_{s_i} \neq \{\}$ *and* $w \notin S_{r_i}$ *and* $w \notin S_{s_i}$ *and*
$Stronger(S_{r_i}, S_{s_i}, T_i) = S_{r_i}$

If $Pr(i+1) = S_{a_i}$ *then either*
(α) $\exists r_i \in R[a_i]$: $S_{r_i} \in Pr(1...i)$ *and* $S_{a_i} = S_{r_i}$ *and*
(α_1) $S_{r_i} = \{\}$ *or*
(α_2) ($\alpha_{2.1}$) $\forall s_i \in R[\neg a_i]$: $w \in S_{s_i}$ *or* $r_i > s_i \in Pr(1...i)$ *and*
($\alpha_{2.2}$) $\forall t_i \in R[a_i]$: $S_{t_i} \in Pr(1...i)$ *and* $Stronger(S_{t_i}, S_{r_i}, T_i) \neq S_{t_i}$ *or*
(β) $S_{a_i} = \{w\}$ *and*
(β_1) $\forall r_i \in R[a_i]$:
($\beta_{1.1}$) $S_{r_i} \in Pr(1...i)$ *and*
($\beta_{1.2}$) $S_{r_i} \neq \{\}$ *and*
($\beta_{1.3}$) $\exists s_i \in R[\neg a_i]$: $S_{s_i} \in Pr(1...i)$ *and* $Stronger(S_{r_i}, S_{s_i}, T_i) \neq S_{r_i}$ *or*
(β_2) $S_{\neg a_i} \in Pr(1...i)$ and $S_{\neg a_i} = \{\}$

$Pr(1...n)$ will contain the supportive sets of all rules and literals in $T_v(P)$, and the required priority relations between all conflicting rules in $T_v(P)$.

In the rest of this section, we prove the equivalence between the distributed theories and the defeasible unified theory $T_v(P)$ (augmented with the priority relations contained in $Pr(1...n)$ following two assumptions:

1. $T_v(P)$ is an acyclic defeasible theory
2. In the case of the distributed theories, there exists a routing mechanism that routes the queries to the appropriate peers (a query about a literal x_i is always routed to P_i, which has defined this literal).

To prove equivalence under these assumptions, we will use the following two theorems:

Theorem 3 *For every literal* x_i,

(a) the set of strict rules in $T_v(P)$ *that support* x_i *(*$R^s[x_i]$*) is the same with the set of local supportive rules* r_i^l *used by P2P_DR to compute* Ans_{x_i}*.*

(b) the set of defeasible rules in $T_v(P)$ *that support* x_i *(*$R^d[x_i]$*) is the same with the set of mapping supportive rules* r_i^m *used by P2P_DR to compute* Ans_{x_i}*.*

(c) (a) and (b) also hold for the rules that contradict x_i

Proof.

(a). The local rules that support x_i and are used by $P2P\_DR$ to compute Ans_{x_i} are those defined in P_i. These rules are represented as strict rules in $T_v(P)$. No other peer theory may contain a local rule that supports x_i, so these rules are the only strict rules that support x_i in $T_v(P)$.

(b). The mapping rules that support x_i and are used by $P2P\_DR$ to compute Ans_{x_i} are those defined in P_i. These rules are represented as defeasible rules in $T_v(P)$. Even if some other peer theory contains a mapping rule that supports x_i, this rule is eliminated during the construction of $T_v(P)$, so P_i's mapping supportive rules are the only defeasible rules that support x_i in $T_v(P)$.

(c) The rules that contradict x_i are in fact the rules that support $\neg x_i$.

So, (a) and (b) also hold for the rules that contradict x_i.

Theorem 4 *If there are no cycles in $T_\upsilon(P)$, P2P_DR will never detect a cycle (and vice versa)*

Proof. In both the defeasible theory $T_\upsilon(P)$ and the distributed theories processed by *P2P_DR*, the evaluation of a query is a sequence of iterative steps, which compute the truth value of a literal, based on the truth value of the literals in the bodies of the rules that support or contradict it. As we have already proved in Theorem 3, the set of rules that are applied in each step are the same. Thus, if there are cycles in the unified theory, *P2P_DR* will also detect the same cycles as well; if not, *P2P_DR* will detect no cycle.

The next theorems concern the relations that hold between the supportive sets of the rules and literals in $Pr(1...n)$ for the unified theory $T_\upsilon(P)$, and the supportive sets of rules and literals of the distributed system nodes, as they are computed by *P2P_DR*.

Theorem 5: *For any literal x_i,*
$localAns_{x_i} = Yes$ *(calculated by local_alg)* $\Leftrightarrow$
$S_{x_i} \in Pr(1...n)$ *and* $S_{x_i} = \{\}$

Left to right proof: Induction on the number of calls of *local_alg*.

Base Case. We will prove that:

(1) If $localAns_{x_i} = Yes$ derives at the first call of *local_alg* in P_i then $S_{x_i} = \{\}$

(1) $localAns_{x_i} = Yes$ derives at the first call of *local_alg* in $P_i \Rightarrow$
$\exists r_i^l \in R_s(x_i)$: $body(r_i^l) = \{\} \Rightarrow$ (using Theorem 3)
$\exists r_i \in T_\upsilon(P)$: $r_i \in R^s[x_i]$ *and* $body(r_i) = \{\} \Rightarrow$
$\exists r_i \in T_\upsilon(P)$: $r_i \in R^s[x_i]$ *and* $S_{r_i} \in Pr(1...n)$ *and* $S_{r_i} = \{\} \Rightarrow$
$S_{x_i} \in Pr(1...n)$ *and* $S_{x_i} = \{\}$

Induction Step. Assume that

(2) $localAns_{x_i} = Yes$ derives during the first n calls of *local_alg* in $P_i \Rightarrow$ $S_{x_i} \in Pr(1...n)$ *and* $S_{x_i} = \{\}$

If $localAns_{x_i} = Yes$ derives in the first $(n+1)$ calls of $local\_alg$ in P_i:

$localAns_{x_i} = Yes \Rightarrow$
$\exists r_i^l \in R_s(x_i)$:
    (α) $body(r_i^l) \neq \{\}$ *and*
    (β) $\forall \alpha \in body(r_i^l)$: $localAns_\alpha = Yes$ (in n calls) $\Rightarrow$ ((2), Theorem 3)

$\exists r_i \in T_\upsilon(P)$:
    (α) $r_i \in R^s[x_i]$ *and* $body(r_i) \neq \{\}$ *and* $S_{r_i} \in Pr(1...n)$ *and*
    (β) $\forall \alpha \in body(r_i)$: $\alpha \in V_i$, $S_\alpha \in Pr(1...n)$ *and* $S_\alpha = \{\} \Rightarrow$
        $S_{x_i} \in Pr(1...n)$ *and* $S_{x_i} = S_{r_i} = \{\}$

Right to left proof: Induction on the derivation steps in $Pr(1...n)$.

Base Case. We will prove that:

(3) $P(2) = S_{x_i} = \{\} \Rightarrow localAns_{x_i} = Yes$
(The supportive set of a literal cannot derive in the first step of the derivation process, unless it contains w)

(3) $P(2) = S_{x_i} = \{\} \Rightarrow$
  $\exists r_i \in T_\upsilon(P)$: $r_i \in R^s[x_i]$ and $S_{r_i} \in P(1)$ *and* $body(r_i) = \{\} \Rightarrow$ (using Theorem 3)
  $\exists r_i^l \in R_s(x_i)$: $body(r_i^l) = \{\} \Rightarrow$
  $localAns_{x_i} = Yes$

Induction Step. Assume that

(4) $S_{x_i} \in P(n)$ *and* $S_{x_i} = \{\} \Rightarrow localAns_{x_i} = Yes$

$S_{x_i} \in P(n+1)$ *and* $S_{x_i} = \{\} \Rightarrow$
$\exists r_i \in R^s[x_i]$: $S_{r_i} \in Pr(1...n)$ *and*
    $\forall \alpha \in body(r_i)$: $\alpha \in V_i$, $S_\alpha \in Pr(1...n)$ *and* $S_\alpha = \{\} \Rightarrow$ ((4), Theorem 3)

$\exists r_i^l \in R_s(x_i)$:
    $\forall \alpha \in body(r_i^l)$: $localAns_{x_i} = Yes \Rightarrow$
    $localAns_{x_i} = Yes$

Theorem 6: *For any literal* x_i,

(a) $Ans_{x_i} = Yes$ *and* $SS_{x_i} = \Sigma \Leftrightarrow S_{x_i} \in Pr(1...n)$ *and* $S_{x_i} = \Sigma$ *and* $w \notin S_{x_i}$
(b) $Ans_{x_i} = No \Leftrightarrow S_{x_i} \in Pr(1...n)$ *and* $w \in S_{x_i}$

Left to Right Proof: Induction on the number of calls of $P2P\_DR$.

Base Case. We will prove that:

(5) If $Ans_{x_i} = Yes$ derives at the first call of $P2P\_DR$ *and* $SS_{x_i} = \Sigma$ then
$S_{x_i} \in Pr(1...n)$ *and* $S_{x_i} = \Sigma$, and

(6) If $Ans_{x_i} = No$ derives at the first call of $P2P\_DR$ then
$S_{x_i} \in Pr(1...n)$ *and* $w \in S_{x_i}$

(5) $Ans_{x_i} = Yes$ derives at the first call of $P2P\_DR \Rightarrow$
$localAns_{x_i} = Yes$ *and* $SS_{x_i} = \{\} \Rightarrow$ (Theorem 5)
$S_{x_i} \in Pr(1...n)$ *and* $S_{x_i} = SS_{x_i} = \{\}$

(6) $Ans_{x_i} = No$ derives at the first call of $P2P\_DR \Rightarrow$
$localAns_{\neg x_i} = Yes$ *or* $\nexists r_i^{lm} \in R_s(x_i) \Rightarrow$ (Theorems 3,5)
$S_{\neg x_i} \in Pr(1...n)$ *and* $S_{\neg x_i} = \{\}$ *or* $\nexists r \in T_\upsilon(P)$: $r \in R^s[x_i] \Rightarrow$
$S_{x_i} \in Pr(1...n)$ *and* $w \in S_{x_i}$

Induction Step. Assume that

(7) $Ans_{x_i} = Yes$ derives in the first n calls of $P2P\_DR$ and $SS_{x_i} = \Sigma \Rightarrow$
$S_{x_i} \in Pr(1...n)$ *and* $S_{x_i} = \Sigma$, and

(8) $Ans_{x_i} = No$ derives in the first n calls of $P2P\_DR \Rightarrow$
$S_{x_i} \in Pr(1...n)$ *and* $w \in S_{x_i}$

If $Ans_{x_i} = Yes$ derives in $(n+1)$ calls of $P2P\_DR$ *and* $SS_{x_i} = \Sigma$:

$SS_{x_i} = \Sigma$ *and* $Ans_{x_i} = Yes \Rightarrow$
(α) $SS_{x_i} = \Sigma$ *and*
(β) $localAns_{x_i} \neq Yes$ *and*
(γ) $localAns_{\neg x_i} \neq Yes$ *and*
(δ) $\exists r_i^{lm} \in R_s(x_i)$:

(δ_1) $SS_{r_i^{lm}} = \Sigma$
(δ_2) $body(r_i^{lm}) \neq \{\}$ *and*
(δ_3) $\forall \alpha \in body(r_i^{lm})$: $Ans_\alpha = Yes$ (in at most n calls) *and*
(δ_4) $\forall s_i^{lm} \in R_c(x_i)$ *either*
($\delta_{4.1}$) $\exists \beta \in body(s_i^{lm})$ s.t. $Ans_\beta = No$ (in at most n calls) *or*
($\delta_{4.2}$) $Stronger(SS_{r_i^{lm}}, SS_{s_i^{lm}}, T_i) = SS_{r_i^{lm}}$ *and*
(δ_5) $\forall t_i^{lm} \in R_s(x_i)$ *either*
($\delta_{5.1}$) $\exists \gamma \in body(t_i^{lm})$ s.t. $Ans_\gamma = No$ (in at most n calls) *or*
($\delta_{5.2}$) $Stronger(SS_{t_i^{lm}}, SS_{r_i^{lm}}, T_i) \neq SS_{t_i^{lm}} \Rightarrow$

(α) $SS_{x_i} = \Sigma$ *and*
(β) $localAns_{x_i} \neq Yes$ *and*
(γ) $localAns_{\neg x_i} \neq Yes$ *and*
(δ) $\exists r_i^{lm} \in R_s(x_i)$:
(δ_1) $SS_{r_i^{lm}} = \Sigma$
(δ_2) $body(r_i^{lm}) \neq \{\}$ *and*
(δ_3) $\forall \alpha \in body(r_i^{lm})$: $Ans_\alpha = Yes$ (in at most n calls) *and*
(δ_4) $\forall s_i^{lm} \in R_c(x_i)$ *either*
($\delta_{4.1}$) $\exists \beta \in body(s_i^{lm})$ s.t. $Ans_\beta = No$ (in at most n calls) *or*
($\delta_{4.2}$) ($\delta_{4.2.1}$) $\forall \beta \in body(s_i^{lm})$: $Ans_\beta = Yes$ (in n calls) *and*
($\delta_{4.2.2}$) $Stronger((\bigcup SS_{\alpha_i}) \cup (\bigcup \alpha_j), (\bigcup SS_{\beta_i}) \cup (\bigcup \beta_j), T_i) = (\bigcup SS_{\alpha_i}) \cup (\bigcup \alpha_j)$
($\forall i, j$: $\alpha_i, \alpha_j \in body(r_i^{lm})$, $\beta_i, \beta_j \in body(s_i^{lm})$, $\alpha_i, \beta_i \in V_i$, $\alpha_j, \beta_j \notin V_i$) *and*
(δ_5) $\forall t_i^{lm} \in R_c(x_i)$ *either*
($\delta_{5.1}$) $\exists \gamma \in body(t_i^{lm})$ s.t. $Ans_\gamma = No$ (in at most n calls) *or*
($\delta_{5.2}$) ($\delta_{5.2.1}$) $\forall \gamma \in body(t_i^{lm})$: $Ans_\gamma = Yes$ (in n calls) *and*
($\delta_{5.2.2}$) $Stronger((\bigcup SS_{\gamma_i}) \cup (\bigcup \gamma_j), (\bigcup SS_{\alpha_i}) \cup (\bigcup \alpha_j), T_i) \neq (\bigcup SS_{\gamma_i}) \cup (\bigcup \gamma_j)$
($\forall i, j$: $\alpha_i, \alpha_j \in body(r_i^{lm})$, $\gamma_i, \gamma_j \in body(t_i^{lm})$, $\alpha_i, \gamma_i \in V_i$, $\alpha_j, \gamma_j \notin V_i$)
$\Rightarrow$ ((7)(8), Theorems 3 and 5)

(α) $SS_{x_i} = \Sigma$ *and*
(β) $S_{x_i} \neq \{\}$ *and*
(γ) $S_{\neg x_i} \neq \{\}$ *and*
(δ) $\exists r_i \in T_\upsilon(P)$: $r \in R^{sd}[x_i]$ *and*
(δ_1) $SS_{r_i} = \Sigma$ *and*
(δ_2) $body(r_i) \neq \{\}$ *and*
(δ_3) $\forall \alpha \in body(r_i)$: $S_\alpha \in Pr(1...n)$ *and* $S_\alpha = SS_\alpha$ *and*
(δ_4) $\forall s_i \in R^{sd}[\neg x_i]$: *either*
($\delta_{4.1}$) $\exists \beta \in body(s_i)$ s.t. $w \in S_\beta$ *or*
($\delta_{4.2}$) ($\delta_{4.2.1}$) $\forall \beta \in body(s_i)$: $S_\beta \in Pr(1...n)$ *and* $S_\beta = SS_\beta$ *and*

$(\delta_{4.2.2})$ $Stronger((\bigcup SS_{\alpha_i}) \cup (\bigcup \alpha_j), (\bigcup SS_{\beta_i}) \cup (\bigcup \beta_j), T_i) = (\bigcup SS_{\alpha_i}) \cup (\bigcup \alpha_j)$

$(\forall i, j\colon\ \alpha_i, \alpha_j \in body(r_i),\ \beta_i, \beta_j \in body(s_i),\ \alpha_i, \beta_i \in V_i,\ \alpha_j, \beta_j \notin V_i)$ *and*

(δ_5) $\forall t_i \in R^{sd}[x_i]$: *either*

$(\delta_{5.1})$ $\exists \gamma \in body(t_i)$ s.t. $w \in S_\gamma$ *or*

$(\delta_{5.2})$ $(\delta_{5.2.1})$ $\forall \gamma \in body(t_i)$: $S_\gamma \in Pr(1...n)$ *and* $S_\gamma = SS_\gamma$ *and*

$(\delta_{5.2.2})$ $Stronger((\bigcup SS_{\gamma_i}) \cup (\bigcup \gamma_j), (\bigcup SS_{\alpha_i}) \cup (\bigcup \alpha_j), T_i) \neq (\bigcup SS_{\gamma_i}) \cup (\bigcup \gamma_j)$

$(\forall i, j\colon\ \alpha_i, \alpha_j \in body(r_i),\ \gamma_i, \gamma_j \in body(t_i),\ \alpha_i, \gamma_i \in V_i,\ \alpha_j, \gamma_j \notin V_i) \Rightarrow$

$S_{x_i} = S_{r_i} = SS_{r_i^{lm}} = \Sigma$

If $Ans_{x_i} = No$ derives in the first $(n+1)$ calls of $P2P\_DR$:

$Ans_{x_i} = No \Rightarrow$

(α) $localAns_{x_i} \neq Yes$ *and*

(β) $localAns_{\neg x_i} \neq Yes$ *and*

(γ) $\forall r_i^{lm} \in R_s(x_i)$ *either*

(γ_1) $\exists \alpha \in body(r_i^{lm})$ s.t. $Ans_\alpha = No$ (in at most n calls) *or*

(γ_2) $\exists s_i^{lm} \in R_c(x_i)$:

$(\gamma_{2.1})$ $body(s_i^{lm}) \neq \{\}$ *and*

$(\gamma_{2.2})$ $\forall \beta \in body(s_i^{lm})$: $Ans_\beta = Yes$ (in at most n calls) *and*

$(\gamma_{2.3})$ $Stronger(SS_{r_i^{lm}}, SS_{s_i^{lm}}, T_i) \neq SS_{r_i^{lm}} \Rightarrow$

(α) $localAns_{x_i} \neq Yes$ *and*

(β) $localAns_{\neg x_i} \neq Yes$ *and*

(γ) $\forall r_i^{lm} \in R_s(x_i)$ *either*

(γ_1) $\exists \alpha \in body(r_i^{lm})$ s.t. $Ans_\alpha = No$ (in at most n calls) *or*

(γ_2) $\exists s_i^{lm} \in R_c(x_i)$:

$(\gamma_{2.1})$ $body(s_i^{lm}) \neq \{\}$ *and*

$(\gamma_{2.2})$ $\forall \beta \in body(s_i^{lm})$: $Ans_\beta = Yes$ (in at most n calls) *and*

$(\gamma_{2.3})$ $Stronger((\bigcup SS_{\alpha_i}) \cup (\bigcup \alpha_j), (\bigcup SS_{\beta_i}) \cup (\bigcup \beta_j), T_i) \neq (\bigcup SS_{\alpha_i}) \cup (\bigcup \alpha_j)$

$(\forall i, j\colon\ \alpha_i, \alpha_j \in body(r_i^{lm}),\ \beta_i, \beta_j \in body(s_i^{lm}),\ \alpha_i, \beta_i \in V_i,\ \alpha_j, \beta_j \notin V_i)$

$\Rightarrow$ ((7)(8), Theorems 3 and 5)

(α) $S_{x_i} \neq \{\}$ *and*

(β) $S_{\neg x_i} \neq \{\}$ *and*

(γ) $\forall r_i \in R^{sd}[x_i]$ *either*

(γ_1) $\exists \alpha \in body(r_i)$ s.t. $S_\alpha \in Pr(1...n)$ *and* $w \in S_\alpha$ *or*

(γ_2) $\exists s_i \in T_v(P)$: $s_i \in R^{sd}[\neg x_i]$ *and*

$(\gamma_{2.1})$ $body(s_i) \neq \{\}$ *and*

$(\gamma_{2.2})$ $\forall \beta \in body(s_i)$: $S_\beta \in Pr(1...n)$ *and* $S_\beta = SS_\beta$ *and*
$(\gamma_{2.3})$ $Stronger((\bigcup SS_{\alpha_i}) \cup (\bigcup \alpha_j), (\bigcup SS_{\beta_i}) \cup (\bigcup \beta_j), T_i) \neq (\bigcup SS_{\alpha_i}) \cup (\bigcup \alpha_j)$
$(\forall i, j$: $\alpha_i, \alpha_j \in body(r_i)$, $\beta_i, \beta_j \in body(s_i)$, $\alpha_i, \beta_i \in V_i$, $\alpha_j, \beta_j \notin V_i) \Rightarrow$
$w \in S_{x_i}$

Right to Left Proof: Induction on the derivation steps in $Pr(1...n)$.

Base Case. We will prove that:

(9) $P(2) = S_{x_i} = \Sigma$ *and* $w \notin \Sigma \Rightarrow Ans_{x_i} = Yes$ *and* $SS_{x_i} = T$, *and*

(10) $P(1) = S_{x_i} = \Sigma$ *and* $w \in \Sigma \Rightarrow Ans_{x_i} = No$

(The supportive set of a literal cannot derive in the first step of the derivation process, unless it contains w)

(9) $P(2) = S_{x_i} = \Sigma$ *and* $w \notin \Sigma \Rightarrow$
$S_{x_i} = \{\} \Rightarrow$ (Theorem 5)
$localAns_{x_i} = Yes \Rightarrow Ans_{x_i} = Yes$

(10) $P(1) = S_{x_i} = \Sigma$ *and* $w \in \Sigma \Rightarrow$
$\nexists r_i \in T_\upsilon(P)$: $r_i \in R^s[x_i] \Rightarrow$ (Theorem 3)
$\nexists r_i^{lm} \in R_s(x_i) \Rightarrow$
$Ans_{x_i} = No$

Induction Step. Assume that

(11) $\Sigma = S_{x_i} \in P(n)$ *and* $w \notin \Sigma \Rightarrow$
$Ans_{x_i} = Yes$ *and* $SS_{x_i} = \Sigma$, *and*

(12) $\Sigma = S_{x_i} \in P(n)$ *and* $w \in \Sigma \Rightarrow$
$Ans_{x_i} = No$

$S_{x_i} = \Sigma \in Pr(n+1)$ *and* $w \notin \Sigma \Rightarrow$
$\exists r_i \in T_\upsilon(P)$: $S_{r_i} = \Sigma \in Pr(1...n)$ *and* $r_i \in R^{sd}[x_i]$ *and* $w \notin \Sigma$ *and either*
(α) $\Sigma = \{\}$ *or*
(β) (β_1) $\forall s_i \in R^{sd}[\neg x_i]$:

$S_{s_i} \in Pr(1...n)$ *and* $Stronger(S_{r_i}, S_{s_i}, T_i) = S_{r_i}$ *and*
(β_2) $\forall t_i \in R^{sd}[x_i]$:
$S_{t_i} \in Pr(1...n)$ *and* $Stronger(S_{t_i}, S_{r_i}, T_i) \neq S_{t_i} \Rightarrow$

$\exists r_i \in T_\upsilon(P)$: $S_{r_i} = \Sigma \in Pr(1...n)$ *and* $r_i \in R^{sd}[x_i]$ *and*
$\forall \alpha \in body(r_i)$: $S_\alpha \in Pr(1...n)$ *and* $w \notin S_\alpha$ *and either*
(α) $\Sigma = \{\}$ *or*
(β) (β_1) $\forall s_i \in R^{sd}[\neg x_i]$:
$(\beta_{1.1})$ $S_{s_i} \in Pr(1...n)$ *and*
$(\beta_{1.2})$ $\forall \beta \in body(s_i), S_\beta \in Pr(1...n)$ *and either*
$(\beta_{1.3})$ $(\beta_{1.3.1})$ $\forall \beta$: $w \notin S_\beta$ *and*
$Stronger((\bigcup S_{\alpha_i}) \cup (\bigcup \alpha_j), (\bigcup S_{\beta_i}) \cup (\bigcup \beta_j), T_i) = (\bigcup S_{\alpha_i}) \cup (\bigcup \alpha_j)$
$(\forall i, j$: $\alpha_i, \alpha_j \in body(r_i)$, $\beta_i, \beta_j \in body(s_i)$, $\alpha_i, \beta_i \in V_i$, $\alpha_j, \beta_j \notin V_i)$ *or*
$(\beta_{1.3.2})$ $\exists \beta$ s.t. $S_\beta \in Pr(1...n)$ *and* $w \in S_\beta$ *and*
(β_2) $\forall t_i \in R^{sd}[\neg x_i]$:
$(\beta_{2.1})$ $S_{t_i} \in Pr(1...n)$ *and*
$(\beta_{2.2})$ $\forall \gamma \in body(t_i)$, $S_\gamma \in Pr(1...n)$ *and either*
$(\beta_{2.3})$ $(\beta_{2.3.1})$ $\forall \gamma$: $w \notin S_\gamma$ *and*
$Stronger((\bigcup SS_{\gamma_i}) \cup (\bigcup \gamma_j), (\bigcup S_{\alpha_i}) \cup (\bigcup \alpha_j), T_i) \neq (\bigcup S_{\gamma_i}) \cup (\bigcup \gamma_j)$
$(\forall i, j$: $\alpha_i, \alpha_j \in body(r_i)$, $\gamma_i, \gamma_j \in body(t_i)$, $\alpha_i, \gamma_i \in V_i$, $\alpha_j, \gamma_j \notin V_i)$ *or*
$(\beta_{2.3.2})$ $\exists \gamma$ s.t. $S_\gamma \in Pr(1...n)$ *and* $w \in S_\gamma \Rightarrow$ ((11),(12), Theorems 3,5)

$\exists r_i^{lm}$: $r_i^{lm} \in R_s(x_i)$ *and* $S_{r_i^{lm}} = \Sigma$ *and*
$\forall \alpha \in body(r_i^{lm})$: $Ans_\alpha = Yes$ *and* $SS_\alpha = S_\alpha$ *and either*
(α) $localAns_{x_i} = Yes$ *or*
(β) (β_1) $\forall s_i^{lm} \in R_c(x_i)$:
$(\beta_{1.1})$ $\forall \beta \in body(s_i^{lm})$: $Ans_\beta = Yes$ *and* $SS_\beta = S_\beta$ *and*
$Stronger((\bigcup SS_{\alpha_i}) \cup (\bigcup \alpha_j), (\bigcup SS_{\beta_i}) \cup (\bigcup \beta_j), T_i) = (\bigcup SS_{\alpha_i}) \cup (\bigcup \alpha_j)$
$(\forall i, j$: $\alpha_i, \alpha_j \in body(r_i^{lm})$, $\beta_i, \beta_j \in body(s_i^{lm})$, $\alpha_i, \beta_i \in V_i$, $\alpha_j, \beta_j \notin V_i)$ *or*
$(\beta_{1.2})$ $\exists \beta \in body(s_i^{lm})$ s.t. $Ans_\beta = No$ *and*
(β_2) $\forall t_i^{lm} \in R_s(x_i)$:
$\forall \gamma \in body(t_i^{lm})$: $Ans_\gamma = Yes$ *and* $SS_\gamma = S_\gamma$ *and*
$Stronger((\bigcup SS_{\gamma_i}) \cup (\bigcup \gamma_j), (\bigcup SS_{\alpha_i}) \cup (\bigcup \alpha_j), T_i) \neq (\bigcup SS_{\gamma_i}) \cup (\bigcup \gamma_j)$
$(\forall i, j$: $\alpha_i, \alpha_j \in body(r_i^{lm})$, $\gamma_i, \gamma_j \in body(t_i^{lm})$, $\alpha_i, \gamma_i \in V_i$, $\alpha_j, \gamma_j \notin V_i)$ *or*
$(\beta_{2.2})$ $\exists \gamma \in body(t_i^{lm})$ s.t. $Ans_\gamma = No \Rightarrow$

$Ans_{x_i} = Yes$ *and* $SS_{x_i} = S_{x_i} = \Sigma$

$S_{x_i} = \Sigma \in Pr(n+1)$ *and* $w \in \Sigma \Rightarrow$

(α) $\forall r_i \in T_\upsilon(P)$ s.t. $r_i \in R^{sd}[x_i]$: $S_{r_i} \in Pr(1...n)$ *and* $body(r_i) \neq \{\}$ *and either*
(α_1) $\exists \alpha \in body(r_i)$: $S_\alpha \in Pr(1...n)$ *and* $w \in S_\alpha$ *or*
(α_2) $\exists s_i \in R^{sd}[\neg x]$: $S_{s_i} \in Pr(1...n)$ *and* $Stronger(S_{r_i}, S_{s_i}, T_i) \neq S_{r_i}$ *or*
(β) $S_{\neg x_i} \in Pr(1...n)$ *and* $S_{\neg x_i} = \{\} \Rightarrow$

(α) $\forall r_i \in T_\upsilon(P) \bigcap R^{sd}[x_i]$: $body(r_i) \neq \{\}$ *and* $\forall \alpha \in body(r_i)$: $S_\alpha \in Pr(1...n)$ *and either*
(α_1) $\exists \alpha \in body(r_i)$: $w \in S_\alpha$ *or*
(α_2) $\exists s_i \in R^{sd}[\neg x_i]$: $\forall \beta \in body(s_i)$: $S_\beta \in Pr(1...n)$ *and*
$(\alpha_{2.1})$ $\forall \alpha \in body(r_i)$: $w \notin S_\alpha$ *and* $\forall \beta \in body(s_i)$: $w \notin S_\beta$ *and*
$(\alpha_{2.2})$ $Stronger((\bigcup S_{\alpha_i}) \cup (\bigcup \alpha_j), (\bigcup S_{\beta_i}) \cup (\bigcup \beta_j), T_i) \neq (\bigcup S_{\alpha_i}) \cup (\bigcup \alpha_j)$
$(\forall i, j$: $\alpha_i, \alpha_j \in body(r_i)$, $\beta_i, \beta_j \in body(s_i)$, $\alpha_i, \beta_i \in V_i$, $\alpha_j, \beta_j \notin V_i)$ *or*
(β) $S_{\neg x_i} \in Pr(1...n)$ *and* $S_{\neg x_i} = \{\} \Rightarrow$ ((11),(12), Theorems 3,5)

(α) $\forall r_i^{lm} \in R_s(x_i)$: $body(r_i^{lm}) \neq \{\}$ *and either*
(α_1) $\exists \alpha \in body(r_i^{lm})$: $Ans_\alpha = No$ *or*
(α_2) $\exists s_i^{lm} \in R_c(x_i)$ *and*
$(\alpha_{2.1})$ $\forall \alpha \in body(r_i^{lm}), \beta \in body(s_i^{lm})$:
$Ans_\alpha = Yes$ *and* $SS_\alpha = S_\alpha$ *and* $Ans_\beta = Yes$ *and* $SS_\beta = S_\beta$ *and*
$(\alpha_{2.2})$ $Stronger((\bigcup S_{\alpha_i}) \cup (\bigcup \alpha_j), (\bigcup S_{\beta_i}) \cup (\bigcup \beta_j), T_i) \neq (\bigcup S_{\alpha_i}) \cup (\bigcup \alpha_j)$
$(\forall i, j$: $\alpha_i, \alpha_j \in body(r_i^{lm})$, $\beta_i, \beta_j \in body(s_i^{lm})$, $\alpha_i, \beta_i \in V_i$, $\alpha_j, \beta_j \notin V_i)$ *or*
(β) $S_{\neg x_i} \in Pr(1...n)$ *and* $S_{\neg x_i} = \{\} \Rightarrow$

$Ans_{x_i} = No$

Using Theorem 6, it is straightforward to prove the following Lemma:

Lemma 7: *For any literal x_i for which,*
$localAns_{x_i} = No$ *and* $localAns_{\neg x_i} = No$
and for any two local or mapping rules $r_i^{lm} \in R_s(x_i)$, $s_i^{lm} \in R_c(x_i)$ for which
$\forall a \in body(r_i^{lm}), \beta \in body(r_i^{lm})$: $Ans_a = Ans_\beta = Yes$
and for their corresponding rules $r_i, s_i \in T_\upsilon(P)$:

$$Stronger(SS_{r_i^{lm}}, SS_{s_i^{lm}}, T_i) = SS_{r_i^{lm}} \Leftrightarrow r_i > s_i \in Pr(1...n)$$

What we need to prove now is that, under the two assumptions described previously, the conclusions that derive from $T_\upsilon(P)$ based on the DL Proof Theory are also returned by $P2P\_DR$, and vice versa. $T_\upsilon(P)$ is a defeasible theory that contains strict, defeasible rules and priorities between conflicting rules, but no facts. A conclusion of $T_v(P)$ is a tagged literal and can have one of the following four forms:

1. $+\Delta x_i$ which is intended to mean that x_i can be definitely proved $T_v(P)$
2. $-\Delta x_i$ which is intended to mean that x_i cannot be definitely proved in $T_v(P)$
3. $+\partial x_i$ which is intended to mean that x_i can be defeasibly proved $T_v(P)$
4. $-\partial x_i$ which is intended to mean that x_i cannot be defeasibly proved in $T_v(P)$

Provability in DL is based on the concept of a derivation in the defeasible theory in $D = (F, R, >)$, where F is the set of facts in D, R denotes the set rules, and $>$ the priority relation on R. A derivation is a finite sequence $P = (P(1), , P(n))$ of tagged literals satisfying the following conditions ($P(1..i)$ denotes the initial part of the sequence P of length i, $R^s[q]$ the set of strict rules that support q and $R^d[q]$ the set of defeasible rules that support q):

$+\Delta$: *If* $P(i+1) = +\Delta q$ *then either*
$q \in F$ *or*
$\exists r \in R^s[q] \forall \alpha \in body(r)$: $+\Delta\alpha \in P(1...i)$

$-\Delta$: *If* $P(i+1) = +\Delta q$ *then either*
$q \notin F$ *and*
$\forall r \in R^s[q] \exists \alpha \in body(r)$: $-\Delta\alpha \in P(1...i)$

$+\partial$: *If* $P(i+1) = +\partial q$ *then either*
(1) $+\Delta q \in P(1...i)$ *or*
(2) (2.1) $\exists r \in R^{sd}[q] \forall \alpha \in body(r)$: $+\partial\alpha \in P(1...i)$ *and*
(2.2) $-\Delta\neg q \in P(1...i)$ *and*
(2.3) $\forall s \in R[\neg q]$
(2.3.1) $\exists \alpha \in body(s)$: $-\partial\alpha \in P(1...i)$ *or*
(2.3.2) $\exists t \in R^{sd}[q]$:
$\forall \alpha \in body(t)$: $+\partial\alpha \in P(1...i)$ *and* $t > s$

$-\partial$: *If* $P(i+1) = -\partial q$ *then*
(1) $-\Delta q \in P(1...i)$ *and*
(2) (2.1) $\forall r \in R^{sd}[q] \exists \alpha \in body(r)$: $-\partial\alpha \in P(1...i)$ *or*
(2.2) $+\Delta\neg q \in P(1...i)$ *or*
(2.3) $\exists s \in R[\neg q]$ *such that*
(2.3.1) $\forall \alpha \in body(s)$: $+\partial\alpha \in P(1...i)$ *and*
(2.3.2) $\forall t \in R^{sd}[q]$ *either*
$\exists \alpha \in body(t)$: $-\partial\alpha \in P(1...i)$ *or* $t \not> s$

We should note that the distributed theories P_i contain no facts. Factual knowledge is expressed in terms of rules with an empty body. Consequently, $T_\upsilon(P)$ also contains no facts. Therefore definite provability in $T_\upsilon(P)$ can be defined as follows:

$+\Delta$: *If* $P(i+1) = +\Delta q$ *then*
$\exists r \in R^s[q] \forall \alpha \in body(r)$: $+\Delta\alpha \in P(1...i)$

$-\Delta$: *If* $P(i+1) = +\Delta q$ *then*
$\forall r \in R^s[q] \exists \alpha \in body(r)$: $-\Delta\alpha \in P(1...i)$

Theorem 8: $T_\upsilon(P) \vdash +\Delta x_i$ *is equivalent to* $localAns_{x_i} = Yes$ *and*
$T_\upsilon(P) \vdash -\Delta x_i$ *is equivalent to* $localAns_{x_i} = No$

Proof. Theorem 8 states that:

(a) If a positive (or negative) definite proof about a literal x_i derives from $T_\upsilon(P)$, then given a query about x_i, $P2P\_DR$ returns $localAns_{x_i} = Yes$ (or $localAns_{x_i} = No$)

(b) vice versa

We can prove $8.a$ using Induction on the number of proof derivation steps in $T_\upsilon(P)$.

Base Case. We will prove that:

(13) $P(1) = +\Delta x_i \Rightarrow localAns_{x_i} = Yes$, and

(14) $P(1) = -\Delta x_i \Rightarrow localAns_{x_i} = No$

(13) $P(1) = +\Delta x_i \Rightarrow$
$\exists r_i \in R^s[x_i]$ s.t. $\forall \alpha \in body(r_i)$: $+\Delta\alpha \in P(0) \Rightarrow$
$\exists r_i \in R^s[x_i]$ s.t. $body(r_i) = \{\} \Rightarrow$ (using Theorem 3)
$\exists r_i^l \in R_s(x_i)$: $body(r_i^l) = \{\} \Rightarrow localAns_{x_i} = Yes$

(14) $P(1) = -\Delta x_i \Rightarrow$
$\forall r_i \in R^s[x_i]$: $\exists \alpha \in body(r)$ s.t. $-\Delta\alpha \in P(0) \Rightarrow$
$\nexists r_i \in R^s[x_i] \Rightarrow$ (using Theorem 3)
$\nexists r_i^l \in R_s(x_i) \Rightarrow localAns_{x_i} = No$

Induction Step. Assume that

(15) $+\Delta x_i \in P(1...n) \Rightarrow localAns_{x_i} = Yes$, and

(16) $-\Delta x_i \in P(1...n) \Rightarrow localAns_{x_i} = No$

For $i = n + 1$

$+\Delta x_i \in P(1...n+1) \Rightarrow$
$\exists r \in R^s[x_i]$: $\forall \alpha \in body(r)$: $+\Delta\alpha \in P(1...n) \Rightarrow$ (using (15) and Theorem 3)
$\exists r_i^l \in R_s(x_i)$ s.t. $\forall \alpha \in body(r_i^l)$: $localAns_\alpha = Yes$
$\Rightarrow localAns_{x_i} = Yes$

$-\Delta x_i \in P(1...n+1) \Rightarrow$
$\forall r \in R^s[x_i]$: $\exists \alpha \in body(r)$: $-\Delta\alpha \in P(1...n) \Rightarrow$ (using (16) and Theorem 3)
$\forall r_i^l \in R_s(x_i)$: $\exists \alpha \in body(r_i^l)$: $localAns_\alpha = No$
$\Rightarrow localAns_{x_i} = No$

We will now prove 8.b using Induction on the number of calls of *local_alg* that are required to compute a local answer for a literal x_i.

Base Case. We will prove that:

(17) If $localAns_{x_i} = Yes$ derives at the first call of *local_alg* in P_i then $T_v(P) \vdash +\Delta x_i$, and

(18) If $localAns_{x_i} = No$ derives at the first call of *local_alg* in P_i then $T_v(P) \vdash -\Delta x_i$

(17) $localAns_{x_i} = Yes$ derives at the first call of *local_alg* in $P_i \Rightarrow$
$\exists r_i^l \in R_s(x_i)$: $body(r_i^l) = \{\} \Rightarrow$ (using Theorem 3)
$\exists r \in T_\upsilon(P)$: $r \in R^s[x_i]$ and $body(r) = \{\} \Rightarrow$
$T_\upsilon(P) \vdash +\Delta x_i$

(18) $localAns_{x_i} = No$ derives at the first call of *local_alg* in $P_i \Rightarrow$
$\nexists r_i^l \in R_s(x_i)$ *and* $\exists s_i^l \in R_c(x_i)$: $body(s_i^l) = \{\} \Rightarrow$ (using Theorem 3)
$\nexists r \in T_\upsilon(P)$: $r \in R^s[x_i]$ *and* $\exists s \in T_\upsilon(P)$: $s \in R_c[x_i]$ and $body(s) = \{\} \Rightarrow$
$T_\upsilon(P) \vdash -\Delta x_i$

Induction Step. Assume that

(19) $localAns_{x_i} = Yes$ derives in the first n calls of *local_alg* in $P_i \Rightarrow$
$T_\upsilon(P) \vdash +\Delta x_i$, and

(20) $localAns_{x_i} = No$ derives in the first n calls of *local_alg* in $P_i \Rightarrow$
$T_\upsilon(P) \vdash -\Delta x_i$

If $localAns_{x_i} = Yes$ derives in $(n+1)$ calls of *local_alg* in P_i:

$localAns_{x_i} = Yes \Rightarrow$
$\exists r_i^l \in R_s(x_i)$: $body(r_i^l) \neq \{\}$ and
$\forall \alpha \in body(r_i^l)$: $localAns_\alpha = Yes$ (in n calls) $\Rightarrow$ ((19), Theorem 3)

$\exists r \in T_\upsilon(P)$: $r \in R^s[x_i]$ and $body(r) \neq \{\}$ and
$\forall \alpha \in body(r)$: $+\Delta\alpha \Rightarrow +\Delta x_i$

If $localAns_{x_i} = No$ derives in $(n+1)$ calls of *local_alg* in P_i:

$localAns_{x_i} = No \Rightarrow$
$\forall r_i^l \in R_s(x_i)$: $body(r_i^l) \neq \{\}$ *and*
$\exists \alpha \in body(r_i^l)$ s.t. $localAns_\alpha = No$ (in n calls) $\Rightarrow$ ((20), Theorem 3)

$\forall r \in T_\upsilon(P)$ s.t. $r \in R^s[x_i]$: $body(r) \neq \{\}$ and
$\exists \alpha \in body(r)$: $-\Delta\alpha \Rightarrow -\Delta x_i$

Theorem 9: $T_\upsilon(P) \vdash +\partial x_i$ *is equivalent to* $Ans_{x_i} = Yes$ *and*
$T_\upsilon(P) \vdash -\partial x_i$ *is equivalent to* $Ans_{x_i} = No$

Proof. Theorem 9 states that:

(a) If a positive (or negative) defeasible proof about a literal x_i derives from $T_\upsilon(P)$, then given a query about x_i, $P2P\_DR$ returns $Ans_{x_i} = Yes$ (or $Ans_{x_i} = No$)

(b) vice versa

We can prove $9.a$ using Induction on the number of proof derivation steps in $T_\upsilon(P)$. A defeasible proof about a literal q cannot derive in one step, as even if there is only one supportive defeasible rule with empty body, in order to prove $+\partial q$, we should priorly derive $-\Delta\neg q$. So the base of the induction will be the first two steps of the derivation process. Furthermore, we should note that there are no defeasible rules with empty body in $T_\upsilon(P)$, as the mapping rules in the distributed theories have (by definition) a non-empty body.

Base Case. We will prove that:

(21) $P(2) = +\partial x_i \Rightarrow Ans_{x_i} = Yes$, and

(22) $P(2) = -\partial x_i \Rightarrow Ans_{x_i} = No$

(21) $P(2) = +\partial x_i \Rightarrow$
(α) $P(1) = +\Delta x_i$ *or*
(β) (β_1) $\exists r \in R^{sd}[x_i]$: $body(r) = \{\}$ *and*
(β_2) $P(1) = -\Delta\neg x_i$ *and*
(β_3) $\nexists s \in R[\neg x_i] \Rightarrow$ (Theorems 3 and 8)

(α) $localAns_{x_i} = Yes$ *or*
(β) (β_1) $\exists r_i^{lm} \in R_s(x_i)$: $body(r_i^{lm}) = \{\}$ *and*
(β_2) $localAns_{\neg x_i} = No$ *and*
(β_3) $\nexists s_i^{lm} \in R_c(x_i) \Rightarrow$

(α) $Ans_{x_i} = Yes$ *or*
(β) (β_1) $\exists r_i^{lm} \in R_s(x_i)$: $body(r_i^{lm}) = \{\}$ *and*
(β_2) $localAns_{\neg x_i} = No$ *and*
(β_3) $CS_{x_i} = \{\} \Rightarrow$
$Ans_{x_i} = Yes$

(22) $P(2) = -\partial x_i \Rightarrow$

(α) $P(1) = -\Delta x_i$ *and*

(β) (β_1) $\nexists r \in R^{sd}[x_i] or$

(β_2) $P(1) = +\Delta \neg x_i \Rightarrow$ (Theorems 3 and 8)

(α) $localAns_{x_i} = No$ *and*

(β) (β_1) $\nexists r_i^{lm} \in R_s(x_i)$ *or*

(β_2) $localAns_{\neg x_i} = Yes \Rightarrow$

(α) $localAns_{x_i} = No$ *and*

(β) (β_1) $SS_{x_i} = \{\}$ *or*

(β_2) $localAns_{\neg x_i} = Yes \Rightarrow$

$Ans_{x_i} = No$

Induction Step. Assume that

(23) $+\partial x_i \in P(1...n) \Rightarrow Ans_{x_i} = Yes$, and

(24) $-\partial x_i \in P(1...n) \Rightarrow Ans_{x_i} = No$

For $i = n + 1$

$+\partial x_i \in P(1...n + 1) \Rightarrow$

(α) $+\Delta x_i \in P(1...n)$ *or*

(β) (β_1) $\exists r \in R^{sd}[x_i]$ s.t. $\forall \alpha \in body(r)$: $+\partial \alpha \in P(1...n)$ *and*

(β_2) $-\Delta \neg x_i \in P(1...n)$ *and*

(β_3) $\forall s \in R^{sd}[\neg x_i]$

$(\beta_{3.1})$ $\exists \beta \in body(s)$: $-\partial \beta \in P(1...n)$ *or*

$(\beta_{3.2})$ $\exists t \in R^{sd}[x_i]$:

$\forall \gamma \in body(t)$: $+\partial \gamma \in P(1...n)$ *and* $t > s \Rightarrow$ ((23),(24), Theorems 3,8, Lemma 7)

(α) $localAns_{x_i} = Yes$ *or*

(β) (β_1) $\exists r_i^{lm} \in R_s(x_i)$ s.t. $\forall \alpha \in body(r_i^{lm})$: $Ans_\alpha = Yes$ *and*

(β_2) $localAns_{\neg x_i} = No$ *and*

(β_3) $\forall s_i^{lm} \in R_c(x_i)$

$(\beta_{3.1})$ $\exists \beta \in body(s_i^{lm})$: $Ans_\beta = No$ *or*

$(\beta_{3.2})$ $\exists t_i^{lm} \in R_s(x_i)$:

$\forall \gamma \in body(t_i^{lm})$: $Ans_\gamma = Yes$ *and*

$Stronger(SS_{t_i^{lm}}, SS_{s_i^{lm}}, T_i) = SS_{t_i^{lm}} \Rightarrow$

$Ans_{x_i} = Yes$

For negative provability:

$-\partial x_i \in P(1...n+1) \Rightarrow$
(α) $-\Delta x_i \in P(1...n)$ *and*
(β) (β_1) $\forall r \in R^{sd}[x_i]$: $\exists \alpha \in body(r)$: $-\partial \alpha \in P(1...n)$ *or*
(β_2) $+\Delta \neg x_i \in P(1...n)$ *or*
(β_3) $\exists s \in R^{sd}[\neg x_i]$ s.t.
$(\beta_{3.1})$ $\forall \beta \in body(s)$: $+\partial \beta \in P(1...n)$ *and*
$(\beta_{3.2})$ $\forall t \in R^{sd}[x_i]$:
$\exists \gamma \in body(t)$: $-\partial \gamma \in P(1...n)$ *or* $t \not> s \Rightarrow$ ((23),(24), Theorems 3,8, Lemma 7)

(α) $localAns_{x_i} = No$ *and*
(β) (β_1) $\forall r_i^{lm} \in R_s(x_i)$: $\exists \alpha \in body(r)$: $Ans_\alpha = No$ *or*
(β_2) $localAns_{\neg x_i} = Yes$ *or*
(β_3) $\exists s_i^{lm} \in R_c(x_i)$:
$(\beta_{3.1})$ $\forall \beta \in body(s_i^{lm})$: $Ans_\beta = Yes$ *and*
$(\beta_{3.2})$ $\forall t_i^{lm} \in R_s(x_i)$:
$\exists \gamma \in body(t_i^{lm})$: $Ans_\gamma = No$ *or*
$Stronger(SS_{t_i^{lm}}, SS_{s_i^{lm}}, T_i) \neq SS_{t_i^{lm}} \Rightarrow$

$Ans_{x_i} = No$

We will now prove 9.b using Induction on the number of calls of $P2P\_DR$ that are required to compute an answer for a literal x_i.

Base Case. We will prove that:

(25) If $Ans_{x_i} = Yes$ derives at the first call of $P2P\_DR$ then
$T_v(P) \vdash +\partial x_i$, and

(26) If $Ans_{x_i} = No$ derives at the first call of $P2P\_DR$ then
$T_v(P) \vdash -\partial x_i$

(25) $Ans_{x_i} = Yes$ derives at the first call of $P2P\_DR \Rightarrow$
$localAns_{x_i} = Yes \Rightarrow$ (Theorem 8)

$T_\upsilon(P) \vdash +\Delta x_i \Rightarrow$
$T_\upsilon(P) \vdash +\partial x_i$

(26) $Ans_{x_i} = No$ derives at the first call of $P2P\_DR \Rightarrow$
$localAns_{\neg x_i} = Yes$ *or* $\nexists r_i^{lm} \in R_s(x_i) \Rightarrow$ (Theorems 3,8)
$T_\upsilon(P) \vdash +\partial\neg x_i$ *or* $\nexists r \in T_\upsilon(P)$: $r \in R^s[x_i] \Rightarrow$
$T_\upsilon(P) \vdash -\partial x_i$

Induction Step. Assume that

(27) $Ans_{x_i} = Yes$ derives in the first n calls of $P2P\_DR \Rightarrow$
$T_\upsilon(P) \vdash +\partial x_i$, and

(28) $Ans_{x_i} = No$ derives in the first n calls of $P2P\_DR \Rightarrow$
$T_\upsilon(P) \vdash -\partial x_i$

If $Ans_{x_i} = Yes$ derives in $(n + 1)$ calls of $P2P\_DR$:

$Ans_{x_i} = Yes \Rightarrow$
(α) $localAns_{x_i} \neq Yes$ *and*
(β) $localAns_{\neg x_i} \neq Yes$ *and*
(γ) $\exists r_i^{lm} \in R_s(x_i)$:
(γ_1) $body(r_i^{lm}) \neq \{\}$ *and*
(γ_2) $\forall\alpha \in body(r_i^{lm})$: $Ans_\alpha = Yes$ (in n calls) *and*
(δ) $\forall s_i^{lm} \in R_c(x_i)$ *either*
(δ_1) $\exists\beta \in body(s_i^{lm})$ s.t. $Ans_\beta = No$ (in n calls) *or*
(δ_2) $Stronger(SS_{r_i^{lm}}, SS_{s_i^{lm}}, T_i) = SS_{r_i^{lm}} \Rightarrow$ ((27)(28), Theorems 3,8, Lemma 7)

(α) $-\Delta x_i$ *and*
(β) $-\Delta\neg x_i$ *and*
(γ) $\exists r \in T_\upsilon(P)$: $r \in R^{sd}[x_i]$ *and*
(γ_1) $body(r) \neq \{\}$ *and*
(γ_2) $\forall\alpha \in body(r)$: $+\partial\alpha$ *and*
(δ) $\forall s \in R^{sd}[x_i]$ *either*
(δ_1) $\exists\beta \in body(s)$ s.t. $-\partial\beta$ *or*
(δ_2) $r > s \Rightarrow$
$+\partial x_i$

If $Ans_{x_i} = No$ derives in $(n+1)$ calls of $P2P\_DR$:

$Ans_{x_i} = No \Rightarrow$

(α) $localAns_{x_i} \neq Yes$ *and*

(β) $localAns_{\neg x_i} \neq Yes$ *and*

(γ) $\forall r_i^{lm} \in R_s(x_i)$ *either*

(γ_1) $\exists \alpha \in body(r_i^{lm})$ s.t. $Ans_\alpha = No$ (in at most n calls) *or*

(γ_2) $\exists s_i^{lm} \in R_c(x_i)$:

$(\gamma_{2.1})$ $body(s_i^{lm}) \neq \{\}$ *and*

$(\gamma_{2.2})$ $\forall \beta \in body(s_i^{lm})$: $Ans_\beta = Yes$ (in n calls) *and*

$(\gamma_{2.3})$ $Stronger(SS_{r_i^{lm}}, SS_{s_i^{lm}}, T_i) \neq SS_{r_i^{lm}} \Rightarrow$ ((27)(28), Theorems 3,8, Lemma 7)

(α) $-\Delta x_i$ *and*

(β) $-\Delta \neg x_i$ *and*

(γ) $\forall r \in R^{sd}[x_i]$ *either*

(γ_1) $\exists \alpha \in body(r)$ s.t. $-\partial \alpha$ *or*

(γ_2) $\exists s \in T_v(P)$: $s \in R^{sd}[\neg x_i]$ *and*

$(\gamma_{2.1})$ $body(s) \neq \{\}$ *and*

$(\gamma_{2.2})$ $\forall \beta \in body(s)$: $+\partial \beta$ *and*

$(\gamma_{2.3})$ $r \not> s \Rightarrow$

$-\partial x_i$

4 1st Approach with DL Local Theories

In this version, we augment the local theories with defeasible rules and with priority relations that are applied on pairs of defeasible local and mapping rules. These features enable a peer to express uncertainty about part of its local knowledge, and to express trust-based preferences not only in the level of peers but also in the level of mapping rules.

To support these features, the algorithm steps are modified as follows: The first step remains unchanged. During this step, a node (say P_i) attempts to produce an answer for the queried literal (say x_i) based on the strict local rules. Even, if there are defeasible local rules that support or contradict x_i, they are not used in this phase.

The 2^{nd} and 3^d step involve building the supportive sets of the (local/mapping) rules that support or contradict x_i, in the same way with the first version of the algorithm. The only difference here is, that in the end, these steps do not produce a single supportive / conflicting set for x_i, but rather collect all the different rules that can be applied to support / contradict x_i.

The 4^{th} step determines the truth value of the queried literal, based on the supportive / conflicting sets of the rules, which are collected in steps 2 and 3, P_i's trust level order, but also on the priorities in P_i's theory. Specifically, if for each of the rules that can be applied to contradict x_i, there is a *superior* (based on the priority relation) supportive rule, or a non-inferior but *stronger* (based on P_i's trust level order) supportive rule, the algorithm returns a positive answer. In any other case, it returns a negative answer for x_i.

We should also note that the *local_alg* procedure remains unchanged. However, the *Stronger* function must be modified to support cases of empty supportive sets.

4.1 The modified algorithm P2P_DR$_{dl}$

Some new symbolisms that we use in this version are:

r_i^l: a local strict rule of P_i

r_i^d: a local defeasible rule of P_i

r_i^m: a mapping rule of P_i

r_i^{ldm}: a rule (local/mapping) of P_i

SR_{x_i}: the set of rules that can be applied to support x_i

CR_{x_i}: the set of rules that can be applied to contradict x_i

P2P_DR$_{dl}(x_i, P_0, P_i, SS_{x_i}, CS_{x_i}, Hist_{x_i}, Ans_{x_i}, T_i)$

if $\exists r_i^l \in R_s(x_i)$ **then**
  $localHist_{x_i} \leftarrow [x_i]$
  run $local\_alg(x_i, localHist_{x_i}, localAns_{x_i})$
  if $localAns_{x_i} = Yes$ **then**
    $Ans_{x_i} \leftarrow localAns_{x_i}$
    terminate
  end if
end if
if $\exists r_i^l \in R_c(x_i)$ **then**
  $localHist_{x_i} \leftarrow [x_i]$
  run $local\_alg(\neg x_i, localHist_{x_i}, localAns_{\neg x_i})$
  if $localAns_{\neg x_i} = Yes$ **then**
    $Ans_{x_i} \leftarrow \neg localAns_{\neg x_i}$
    terminate
  end if
end if
$SR_{x_i} \leftarrow \{\}$
for all $r_i^{ldm} \in R_s(x_i)$ **do**
  $SS_{r_i} \leftarrow \{\}$
  for all $b_t \in body(r_i^{ldm})$ **do**
    if $b_t \in Hist_{x_i}$ **then**
      stop and check the next rule
    else
      $Hist_{b_t} \leftarrow Hist_{x_i} \cup b_t$
      run $P2P\_DR(b_t, P_i, P_t, SS_{b_t}, CS_{b_t}, Hist_{b_t}, Ans_{b_t}, T_t)$
      if $Ans_{b_t} = No$ **then**
        stop and check the next rule
      else if $Ans_{b_t} = Yes\ and\ b_t \notin V_i$ **then**
        $SS_{r_i} \leftarrow SS_{r_i} \cup b_t$
      else

$SS_{r_i} \leftarrow SS_{r_i} \cup SS_{b_t}$
end if
end if
end for
if $SR_{x_i} = \{\}$ *or* $Stronger(SS_{r_i}, SS_{x_i}, T_i) = SS_{r_i}$ **then**
$SS_{x_i} \leftarrow SS_{r_i}$
end if
$SR_{x_i} \leftarrow SR_{x_i} \cup r_i^{ldm}$
end for
if $SR_{x_i} = \{\}$ **then**
return $Ans_{x_i} = No$ and terminate
end if
$CR_{x_i} \leftarrow \{\}$
for all $r_i^{ldm} \in R_c(x_i)$ **do**
$SS_{r_i} \leftarrow \{\}$
for all $b_t \in body(r_i^{ldm})$ **do**
if $b_t \in Hist_{x_i}$ **then**
stop and check the next rule
else
$Hist_{b_t} \leftarrow Hist_{x_i} \cup b_t$
run $P2P\_DR(b_t, P_i, P_t, SS_{b_t}, CS_{b_t}, Hist_{b_t}, Ans_{b_t}, T_t)$
if $Ans_{b_t} = No$ **then**
stop and check the next rule
else if $Ans_{b_t} = Yes$ *and* $b_t \notin V_i$ **then**
$SS_{r_i} \leftarrow SS_{r_i} \cup b_t$
else
$SS_{r_i} \leftarrow SS_{r_i} \cup SS_{b_t}$
end if
end if
end for
$CR_{x_i} \leftarrow CR_{x_i} \cup r_i^{ldm}$
end for
if $CR_{x_i} = \{\}$ **then**
return $Ans_{x_i} = Yes$ and terminate
end if
for all $r'_i \in CR_{x_i}$ **do**
if $\nexists r_i \in SR_{x_i}$: $r_i > r'_i$ *or* $(r'_i \not> r_i$ *and* $Stronger(SS_{r_i}, SS_{r'_i}, T_i) = SS_{r_i})$ **then**
return $Ans_{x_i} = No$ and terminate
end if

end for
return $Ans_{x_i} = Yes$ and SS_{x_i}

The *Stronger* function is modified as follows:

Stronger(S, C, T_i)
if $S = \{\}$ *and* $C = \{\}$ **then**
    $Stronger = None$
end if
if $S = \{\}$ *and* $C \neq \{\}$ **then**
    $Stronger = S$
end if
if $S \neq \{\}$ *and* $C = \{\}$ **then**
    $Stronger = C$
end if
$a^w \leftarrow a_k \in S$ s.t. *for all* $a_i \in S : P_k$ does not precede P_i in $T_i)$
$b^w \leftarrow a_l \in C$ s.t. *for all* $b_j \in C : P_l$ does not precede P_j in $T_i)$
if P_k precedes P_l in T_i **then**
    $Stronger = S$
else if P_l precedes P_k in T_i **then**
    $Stronger = C$
else
    $Stronger = None$
end if

4.2 Properties of P2P_DR$_{dl}$

4.2.1 Termination and Number of Messages

P2P_DR$_{dl}$ shares the same properties with **P2P_DR** with regard to termination and the total number of messages that need to be exchanged between the system peers for the computation of a single query.

Based on the facts that (α) **P2P_DR**$_{dl}$ terminates either by detecting a cycle or by returning an answer about the truth value of the queried literal, and (β) there are a finite number of nodes, each one with a finite number of literals, and consequently with a finite number of rules and priority relations, **P2P_DR**$_{dl}$ *is guaranteed to terminate.*

In the same way with **P2P_DR**, **P2P_DR**$_{dl}$ requires, in the worst case, each node to check the truth value of all the remote literals that are involved in its mapping rules at most once. We have already proved that this procedure will result in a number of messages that is proportional to the square of the maximum number of acquaintances a system node may have, and in the worst case that each node has defined mappings which involve all the other system nodes, *the total number of messages is* $O(n^2)$ (where n stands for the node population).

4.2.2 Single Node Complexity

Adding defeasible local theories in the system adds an overhead to the computational complexity of the algorithm on a single node. By comparing the two versions, it is obvious that the additional overhead is imposed by building SR_{x_i} and CR_{x_i} (the collections of rules that can be applied to support / contradict x_i) and by the module that checks the *priority* and *strength* relations for each pair of conflicting rules. Building SR_{x_i} (CR_{x_i}) has an overhead which is proportional to the total number of rules that support (contradict) x_i. Consequently, in the worst case that for the computation of the truth value of x_i, all rules in P_i are involved, building these two collections has a total overhead which is proportional to the total number of rules in P_i ($O(n_r)$).

The second module requires for each pair of conflicting rules $r_i \in SR_{x_i}$, $r'_i \in CR_{x_i}$ checking their priority relation and comparing their supportive sets (SS_{r_i}, $SS_{r'_i}$) through the Stronger function. Considering that the number of elements in a Supportive Set is in the worst case $O(n_{ACQ} \times n_l)$, where n_{ACQ} is the number of acquaintances a peer may have, and n_l is the number of literals a peer may define. So, the total overhead of this module $O(n_r^2 \times n_{ACQ} \times n_l)$, where n_r is the number of rules in a peer theory.

Considering that the second module replaces the part of the first version, which compares SS_{x_i} and CS_{x_i} through the *Stronger* function to compute the final answer for x_i, and that all the other parts of the algorithm remain unchanged, the computational complexity in this version is

$$O(n_{rloc} \times n_l^{rloc} \times n_l + n_r \times n_l^r \times n_{ACQ} \times n_l + n_r + n_r^2 \times n_{ACQ} \times n_l)$$

n_{rloc} is the number of local rules defined by a peer

n_r is the number of (local and mapping) rules defined by a peer
n_l^r is the number of literals in the body of a rule
n_l^{rloc} is the number of literals in the body of a local rule
n_l is the number of literals defined by one peer
n_{ACQ} is the number of a peer's acquaintances

Assuming that (a) $n_l^r = O(n_{ACQ} \times n_l)$; and (b) $n_l^{rloc} = O(n_l)$, the overall complexity is

$$O(n_{ACQ}^2 \times n_l^2 \times n_r + n_{ACQ} \times n_l \times n_r^2)$$

In the worst case, that that all peers have defined mappings that involve all the other system nodes: $n_{ACQ} = O(n)$, and the overall complexity is

$$O(n^2 \times n_l^2 \times n_r + n \times n_l \times n_r^2)$$

4.3 Equivalent Unified Defeasible Theory

The steps that we have to take to build an equivalent defeasible theory based on the distributed local theories and the trust level orderings of each system node are four. The second and third steps are exactly the same with the respective steps followed in the case of non-defeasible local theories. The first and fourth steps are modified as follows:

1. The strict local rules and the defeasible rules of each peer's theory are also part of the unified theory, $T_v(P)$ (without any modification).

4. Each priority relation that is part of the local theories is also part of the unified theory, $T_v(P)$. For each pair of conflicting rules, for which there is no priority relation in the local theories, we add a priority relation based on the peers' trust level ordering using the following procedure:

Priorities$_{dl}$

The derivation of priorities between conflicting rules in $T_v(P)$ is a finite sequence $Pr = (Pr(1), ..., Pr(n))$, where each $Pr(i)$ can be one of the followings:

- The supportive set of a rule in $T_v(P)$ (a set of literals).
- A priority relation between two conflicting rules in $T_v(P)$

- The supportive set of a literal in $T_v(P)$ (a set of literals).

Assuming that the first i steps of this derivation have computed $Pr(1...i)$, which is the initial part of the sequence Pr of length i, the next part of this sequence ($Pr(i+1)$) will be either the supportive set of a rule (S_{r_i}), or a priority relation ($r_i > s_i$), or the supportive set of a literal (S_{a_i}).

If $Pr(i+1) = S_{r_i}$ *then either*
(α) S_{r_i} ={s} (*where* s *is the strongest possible element*) *and*
$r_i \in R^s$ *and* $\forall a_i \in body(r_i)$: $S_{a_i} \in Pr(1...i)$ *and* $S_{a_i} = s$ *or*
(β) $S_{r_i} = (\bigcup S_{a_i}) \cup (\bigcup a_j)$, *and*
$\forall a_i$: $a_i \in V_i$, $a_i \in body(r_i)$, $S_{a_i} \in Pr(1...i)$ *and*
$\forall a_j$: $a_j \notin V_i$, $a_j \in body(r_i)$, $S_{a_j} \in Pr(1...i)$, $w \notin S_{a_j}$ *or*
(γ) $S_{r_i} = \{w\}$, *and*
$\exists a_j$, s.t. $a_j \notin V_i$, $a_j \in body(r_i)$, $S_{a_j} \in Pr(1...i)$, $w \in S_{a_j}$

If $Pr(i+1) = r_i > s_i$ *then*
$S_{r_i}, S_{s_i} \in Pr(1...i)$ *and* r_i, s_i *are conflicting and*
$w \notin S_{r_i}$ *and* $w \notin S_{s_i}$ *and*
$r_i > s_i, s_i > r_i \notin P_i$ *and*
$Stronger(S_{r_i}, S_{s_i}, T_i) = S_{r_i}$

If $Pr(i+1) = S_{a_i}$ *then either*
(α) $\exists r_i \in R[a_i]$: $S_{r_i} \in Pr(1...i)$ *and* $S_{a_i} = S_{r_i}$ *and either*
(α_1) $S_{r_i} = \{s\}$ *or*
(α_2) $S_{r_i} \neq \{s\}$ *and*
($\alpha_{2.1}$) $\forall s_i \in R[\neg a_i]$: $\exists q_i \in R[a_i]$ s.t. $S_{q_i} \in Pr(1...i)$ *and*
($\alpha_{2.1.1}$) $w \notin S_{q_i}$ *and* $q_i > s_i \in P_i$ *or*
($\alpha_{2.1.2}$) $q_i > s_i \in Pr(1...i)$ *and*
($\alpha_{2.2}$) $\forall t_i \in R[a_i]$: $S_{t_i} \in Pr(1...i)$ *and* $Stronger(S_{t_i}, S_{r_i}, T_i) \neq S_{t_i}$ *or*
(β) $S_{a_i} = \{w\}$ *and*
(β_1) $\exists s_i \in R[\neg a_i]$:
($\beta_{1.1}$) $S_{s_i} \in Pr(1...i)$ *and*
($\beta_{1.2}$) $\forall r_i \in R[a_i]$ *either*
($\beta_{1.2.1}$) $S_{r_i} \in Pr(1...i)$ *and* $w \in S_{r_i}$ *or*
($\beta_{1.2.2}$) $r_i > s_i \notin P_i$ *and either*
$Stronger(S_{r_i}, S_{s_i}, T_i) \neq S_{r_i}$ *or* $s_i > r_i \in P_i$ *or*
(β_2) $S_{\neg a_i} \in Pr(1...i)$ and $S_{\neg a_i} = \{s\}$

$Pr(1...n)$ will contain the supportive sets of all rules and literals in $T_v(P)$, and the required priority relations between all conflicting rules in $T_v(P)$, for which there is no priority relation in the original local theories.

It is now left to check if Theorems 3-6, Lemma 7, and Theorems 8-9 hold for the relation between the distributed local theories (which are now augmented with defeasible local rules and priorities) and the unified defeasible theory that is constructed in the way that we describe above.

Theorem 3 holds with some small modifications. Specifically, for this new version of the algorithm, it should be modified as follows:

Theorem 3 (**P2P_DR**$_{dl}$) *For every literal* x_i*,*

(a) the set of strict rules in $T_v(P)$ *that support* x_i *(*$R^s[x_i]$*) is the same with the set of local* ***strict*** *supportive rules* r_i^l *used by P2P_DR to compute* Ans_{x_i}*.*

(b) the set of defeasible rules in $T_v(P)$ *that support* x_i *(*$R^d[x_i]$*) derives from the* ***the unification of the sets of local defeasible supportive rules*** r_i^d ***and mapping supportive rules*** r_i^m *used by P2P_DR to compute* Ans_{x_i}*.*

(c) (a) and (b) also hold for the rules that contradict x_i

Proof.

(a). The local strict rules that support x_i and are used by *P2P_DR* to compute Ans_{x_i} are those defined in P_i. These rules are also part of $T_v(P)$. No other peer theory may contain a local strict rule that supports x_i, so these rules are the only strict rules that support x_i in $T_v(P)$.

(b). The local defeasible rules that support x_i and are used by *P2P_DR* to compute Ans_{x_i} are those defined in P_i. These rules are also part of $T_v(P)$. No other peer theory may contain a local strict rule that supports x_i. The mapping rules that support x_i and are used by *P2P_DR* to compute Ans_{x_i} are those defined in P_i. These rules are also represented as defeasible rules in $T_v(P)$. No other peer theory may contain a local defeasible rule that supports x_i, and even if some other peer theory contains a mapping rule that supports x_i, this rule is eliminated during the construction of $T_v(P)$, so P_i's local defeasible and mapping supportive rules are the only defeasible rules that support x_i in $T_v(P)$.

(c) The rules that contradict x_i are in fact the rules that support $\neg x_i$. So, (a) and (b) also hold for the rules that contradict x_i.

Based on Theorem 3 for **P2P_DR**$_{dl}$, we can derive that Theorem 4 holds also for this version (*If there are no cycles in* $T_\upsilon(P)$, *P2P_DR*$_{dl}$ *will never detect a cycle; and vice versa*). The proof is exactly the same with the one that we presented for the case of **P2P_DR**.

Theorem 5 is modified as follows:

Theorem 5 (for **P2P_DR**$_{dl}$): *For any literal* x_i,
$localAns_{x_i} = Yes$ (*calculated by local_alg*) $\Leftrightarrow$
$S_{x_i} \in Pr(1...n)$ *and* $S_{x_i} = \{s\}$

Left to right proof: Induction on the number of calls of *local_alg*.

Base Case. We will prove that:

(1) If $localAns_{x_i} = Yes$ derives at the first call of *local_alg* in P_i then
$S_{x_i} = \{s\}$

(1) $localAns_{x_i} = Yes$ derives at the first call of *local_alg* in $P_i \Rightarrow$
$\exists r_i^l \in R_s(x_i)$: $body(r_i^l) = \{\} \Rightarrow$ (using Theorem 3)
$\exists r_i \in T_\upsilon(P)$: $r_i \in R^s[x_i]$ *and* $body(r_i) = \{\} \Rightarrow$
$\exists r_i \in T_\upsilon(P)$: $r_i \in R^s[x_i]$ *and* $S_{r_i} \in Pr(1...n)$ *and* $S_{r_i} = \{s\} \Rightarrow$
$S_{x_i} \in Pr(1...n)$ *and* $S_{x_i} = \{s\}$

Induction Step. Assume that

(2) $localAns_{x_i} = Yes$ derives during the first n calls of *local_alg* in $P_i \Rightarrow$
$S_{x_i} \in Pr(1...n)$ *and* $S_{x_i} = \{s\}$

If $localAns_{x_i} = Yes$ derives in the first $(n+1)$ calls of *local_alg* in P_i:

$localAns_{x_i} = Yes \Rightarrow$
$\exists r_i^l \in R_s(x_i)$:
(α) $body(r_i^l) \neq \{\}$ *and*
(β) $\forall \alpha \in body(r_i^l)$: $localAns_\alpha = Yes$ (in n calls) $\Rightarrow$ ((2), Theorem 3)

$\exists r_i \in T_\upsilon(P)$:

(α) $r_i \in R^s[x_i]$ *and* $body(r_i) \neq \{\}$ *and* $S_{r_i} \in Pr(1...n)$ *and*

(β) $\forall \alpha \in body(r_i)$: $\alpha \in V_i$, $S_\alpha \in Pr(1...n)$ *and* $S_\alpha = \{s\} \Rightarrow$

$S_{x_i} \in Pr(1...n)$ *and* $S_{x_i} = S_{r_i} = \{s\}$

Right to left proof: Induction on the derivation steps in $Pr(1...n)$.

Base Case. We will prove that:

(3) $P(2) = S_{x_i} = \{s\} \Rightarrow localAns_{x_i} = Yes$
(The supportive set of a literal cannot derive in the first step of the derivation process, unless it contains w)

(3) $P(2) = S_{x_i} = \{s\} \Rightarrow$
$\exists r_i \in T_\upsilon(P)$: $r_i \in R^s[x_i]$ *and* $S_{r_i} \in P(1)$ *and* $S_{r_i} = \{s\} \Rightarrow$
$\exists r_i \in T_\upsilon(P)$: $r_i \in R^s[x_i]$ *and* $S_{r_i} \in P(1)$ *and* $body(r_i) = \{\} \Rightarrow$ (using Theorem 3)
$\exists r_i^l \in R_s(x_i)$: $body(r_i^l) = \{\} \Rightarrow$
$localAns_{x_i} = Yes$

Induction Step. Assume that

(4) $S_{x_i} \in P(n)$ *and* $S_{x_i} = \{s\} \Rightarrow localAns_{x_i} = Yes$

$S_{x_i} \in P(n+1)$ *and* $S_{x_i} = \{s\} \Rightarrow$
$\exists r_i \in R^s[x_i]$: $S_{r_i} \in Pr(1...n)$ *and* $S_{r_i} = \{s\} \Rightarrow$
$\exists r_i \in R^s[x_i]$: $S_{r_i} \in Pr(1...n)$ *and*
$\forall \alpha \in body(r_i)$: $\alpha \in V_i$, $S_\alpha \in Pr(1...n)$ *and* $S_\alpha = \{s\} \Rightarrow$ ((4), Theorem 3)
$\exists r_i^l \in R_s(x_i)$: $\forall \alpha \in body(r_i^l)$: $localAns_{x_i} = Yes \Rightarrow$
$localAns_{x_i} = Yes$

Theorem 6 also holds for this version but with some minor modifications. Specifically:

Theorem 6 (for **P2P_DR**$_{dl}$): *For any literal* x_i, *for which* $localAns_{x_i} = No$,

(a) $Ans_{x_i} = Yes$ *and* $SS_{x_i} = \Sigma \Leftrightarrow S_{x_i} \in Pr(1...n)$ *and* $S_{x_i} = \Sigma$ *and* $w \notin S_{x_i}$

(b) $Ans_{x_i} = No \Leftrightarrow S_{x_i} \in Pr(1...n)$ *and* $w \in S_{x_i}$

Left to Right Proof: Induction on the number of calls of $P2P\_DR$.

Base Case. We will prove that for a literal x_i for which $localAns_{x_i} = No$:

(5) If $Ans_{x_i} = Yes$ derives at the first call of $P2P\_DR_{dl}$ *and* $SS_{x_i} = \Sigma$ then
$S_{x_i} \in Pr(1...n)$ *and* $S_{x_i} = \Sigma$, and

(6) If $Ans_{x_i} = No$ derives at the first call of $P2P\_DR_{dl}$ then
$S_{x_i} \in Pr(1...n)$ *and* $w \in S_{x_i}$

(5) $Ans_{x_i} = Yes$ derives at the first call of $P2P\_DR_{dl}$ *and* $localAns_{x_i} \neq Yes$ *and* $SS_{x_i} = \Sigma \Rightarrow$
(α) $localAns_{x_i} \neq Yes$ *and*
(β) $localAns_{\neg x_i} \neq Yes$ *and*
(γ) $\exists r_i^d \in R_s(x_i)$: $body(r_i^d) = \{\}$ *and* $\Sigma = SS_{x_i} = SS_{r_i}$ *and*
(δ) $\nexists t_i^{ldm} \in R_s(x_i)$: $body(t_i) \neq \{\}$ *and*
(ϵ) $\forall s_i^{dm} \in R_c(x_i)$: $body(s_i) = \{\}$ *and* $\exists t_i^{ldm} \in R_s(x_i)$: $t_i > s_i \Rightarrow$

(α) $localAns_{x_i} \neq Yes$ *and*
(β) $localAns_{\neg x_i} \neq Yes$ *and*
(γ) $\exists r_i^d \in R_s(x_i)$: $body(r_i^d) = \{\}$ *and* $\Sigma = \{\}$ *and*
(δ) $\nexists t_i^{ldm} \in R_s(x_i)$: $body(t_i) \neq \{\}$ *and*
(ϵ) $\forall s_i^{dm} \in R_c(x_i)$: $body(s_i) = \{\}$ *and* $\exists t_i^{ldm} \in R_s(x_i)$: $t_i > s_i \Rightarrow$ (Theorems 3,5)

(α) $S_{x_i} \neq \{s\}$ *and*
(β) $S_{\neg x_i} \neq \{s\}$ *and*
(γ) $\exists r \in T_\upsilon(P)$: $r \in R^d[x_i]$ *and* $body(r) = \{\}$ *and*
(δ) $\nexists t \in T_\upsilon(P)$: $t \in R[x_i]$ *and* $body(t) \neq \{\}$ *and*
(ϵ) $\forall s \in T_\upsilon(P)$ s.t. $s \in R^d[\neg x_i]$: $body(s) = \{\}$ *and* $\exists t \in R[x_i]$: $t > s \Rightarrow$

$S_{x_i} = S_r = \{\} = SS_{x_i} = \Sigma$

(6) $Ans_{x_i} = No$ derives at the first call of $P2P\_DR_{dl} \Rightarrow$
(α) $localAns_{\neg x_i} = Yes$ *or*
(β) $\nexists r_i^{ldm} \in R_s(x_i)$ *or*
(γ) (γ_1) $localAns_{x_i} \neq Yes$ *and*
(γ_2) $localAns_{\neg x_i} \neq Yes$ *and*
(γ_3) $\forall s_i^{ldm} \in R_c(x_i)$: $body(s_i) = \{\}$ *and*
(γ_4) $\forall r_i^{dm} \in R_s(x_i)$: $body(r_i) = \{\}$ *and*
(γ_5) $\exists q_i^{ldm} \in R_c(x_i)$ s.t. $\forall r_i \in R_s(x_i)$: $r_i \not> q_i \Rightarrow$ (Theorems 3,5)

(α) $S_{\neg x_i} = \{s\}$ *or*
(β) $\nexists r \in T_\upsilon(P)$: $r \in R[x_i]$ *or*
(γ) (γ_1) $S_{x_i} \neq \{s\}$ *and*
(γ_2) $S_{\neg x_i} \neq \{s\}$ *and*
(γ_3) $\forall s \in T_\upsilon(P)$ s.t. $s \in R^d[\neg x_i]$: $body(s) = \{\}$ *and*
(γ_4) $\forall r \in T_\upsilon(P)$ s.t. $r \in R^d[x_i]$: $body(r) = \{\}$ *and*
(γ_5) $\exists q \in T_\upsilon(P) \cup R_c(x_i)$ s.t. $\forall r \in T_\upsilon(P) \cup R[x_i]$: $r \not> q \Rightarrow$

$S_{x_i} \in Pr(1...n)$ *and* $w \in S_{x_i}$

Induction Step. Assume that

(7) $Ans_{x_i} = Yes$ derives in the first n calls of $P2P\_DR_{dl}$ and $SS_{x_i} = \Sigma \Rightarrow$
$S_{x_i} \in Pr(1...n)$ *and* $S_{x_i} = \Sigma$, and

(8) $Ans_{x_i} = No$ derives in the first n calls of $P2P\_DR_{dl} \Rightarrow$
$S_{x_i} \in Pr(1...n)$ *and* $w \in S_{x_i}$

If $Ans_{x_i} = Yes$ derives in $(n+1)$ calls of $P2P\_DR_{dl}$ *and* $SS_{x_i} = \Sigma$:

$SS_{x_i} = \Sigma$ *and* $Ans_{x_i} = Yes \Rightarrow$
(α) $SS_{x_i} = \Sigma$ *and*
(β) $localAns_{x_i} \neq Yes$ *and*
(γ) $localAns_{\neg x_i} \neq Yes$ *and*
(δ) $\exists r_i^{ldm} \in SR_{x_i}$:
(δ_1) $SS_{r_i^{ldm}} = \Sigma$ *and*
(δ_2) $\forall t_i^{ldm} \in SR_{x_i}$: $Stronger(SS_{t_i}, SS_{r_i}, T_i) \neq SS_{t_i}$ *and*
(ϵ) $\forall s_i^{ldm} \in CR_{x_i}$ $\exists t_i^{ldm} \in SR_{x_i}$: $Stronger(SS_{t_i}, SS_{s_i}, T_i) = SS_{t_i} \Rightarrow$

(α) $SS_{x_i} = \Sigma$ *and*
(β) $localAns_{x_i} \neq Yes$ *and*
(γ) $localAns_{\neg x_i} \neq Yes$ *and*
(δ) $\exists r_i^{ldm} \in R_s(x_i)$:
(δ_1) $SS_{r_i^{ldm}} = \Sigma$ *and*
(δ_2) $\forall \alpha \in body(r_i^{ldm})$: $Ans_\alpha = Yes$ (in at most n calls) *and*
(δ_3) $\forall t_i^{ldm} \in R_s(x_i)$: *either*
($\delta_{3.1}$) $\exists \gamma \in body(t_i^{ldm})$ s.t. $Ans_\gamma = No$ *or*
($\delta_{3.2}$) $Stronger(SS_{t_i}, SS_{r_i}, T_i) \neq SS_{t_i}$ *and*

(ϵ) $\forall s_i^{ldm} \in R_c(x_i)$ *either*

(ϵ_1) $\exists \beta \in body(s_i^{ldm})$ s.t. $Ans_\beta = No$ *or*

$(\epsilon_2) \exists t_i^{ldm} \in R_s(x_i)$:

$(\epsilon_{2.1})$ $\forall \gamma \in body(t_i^{ldm})$: $Ans_\gamma = Yes$ (in at most n calls) *and*

$(\epsilon_{2.2})$ $Stronger(SS_{t_i}, SS_{s_i}, T_i) = SS_{t_i} \Rightarrow$

(α) $SS_{x_i} = \Sigma$ *and*

(β) $localAns_{x_i} \neq Yes$ *and*

(γ) $localAns_{\neg x_i} \neq Yes$ *and*

(δ) $\exists r_i^{ldm} \in R_s(x_i)$:

(δ_1) $SS_{r_i^{ldm}} = \Sigma$ *and*

(δ_2) $\forall \alpha \in body(r_i^{ldm})$: $Ans_\alpha = Yes$ (in at most n calls) *and*

(δ_3) $\forall t_i^{ldm} \in R_s(x_i)$: *either*

$(\delta_{3.1})$ $\exists \gamma \in body(t_i^{ldm})$ s.t. $Ans_\gamma = No$ *or*

$(\delta_{3.2})$ $(\delta_{3.2.1})$ $\forall \gamma \in body(t_i^{lm})$: $Ans_\gamma = Yes$ (in n calls) *and*

$(\delta_{3.2.2})$ $Stronger((\bigcup SS_{\gamma_i}) \cup (\bigcup \gamma_j), (\bigcup SS_{\alpha_i}) \cup (\bigcup \alpha_j), T_i) \neq (\bigcup SS_{\gamma_i}) \cup (\bigcup \gamma_j)$

$(\forall i, j$: $\alpha_i, \alpha_j \in body(r_i^{ldm})$, $\gamma_i, \gamma_j \in body(t_i^{ldm})$, $\alpha_i, \gamma_i \in V_i$, $\alpha_j, \gamma_j \notin V_i)$

(ϵ) $\forall s_i^{ldm} \in R_c(x_i)$ *either*

(ϵ_1) $\exists \beta \in body(s_i^{ldm})$ s.t. $Ans_\beta = No$ *or*

$(\epsilon_2) \exists t_i^{ldm} \in R_s(x_i)$:

$(\epsilon_{2.1})$ $\forall \gamma \in body(t_i^{ldm})$: $Ans_\gamma = Yes$ (in at most n calls) *and*

$(\epsilon_{2.2})$ $Stronger((\bigcup SS_{\gamma_i}) \cup (\bigcup \gamma_j), (\bigcup SS_{\beta_i}) \cup (\bigcup \beta_j), T_i) = (\bigcup SS_{\gamma_i}) \cup (\bigcup \gamma_j)$

$(\forall i, j$: $\gamma_i, \gamma_j \in body(t_i^{ldm})$, $\beta_i, \beta_j \in body(s_i^{ldm})$, $\gamma_i, \beta_i \in V_i$, $\alpha_j, \beta_j \notin V_i)$

$\Rightarrow$ ((7)(8), Theorems 3 and 5)

(α) $SS_{x_i} = \Sigma$ *and*

(β) $S_{x_i} \neq \{\}$ *and*

(γ) $S_{\neg x_i} \neq \{\}$ *and*

(δ) $\exists r_i \in T_v(P)$: $r_i \in R^{sd}[x_i]$ *and*

(δ_1) $SS_{r_i} = \Sigma$ *and*

(δ_2) $\forall \alpha \in body(r_i)$: $S_\alpha \in Pr(1...n)$ *and* $S_\alpha = SS_\alpha$ *and*

(δ_3) $\forall t_i \in R^{sd}[x_i]$: *either*

$(\delta_{3.1})$ $\exists \gamma \in body(t_i)$ s.t. $w \in S_\gamma$ *or*

$(\delta_{3.2})$ $(\delta_{3.2.1})$ $\forall \gamma \in body(t_i)$: $S_\gamma \in Pr(1...n)$ *and* $S_\gamma = SS_\gamma$ *and*

$(\delta_{3.2.2})$ $Stronger((\bigcup SS_{\gamma_i}) \cup (\bigcup \gamma_j), (\bigcup SS_{\alpha_i}) \cup (\bigcup \alpha_j), T_i) \neq (\bigcup SS_{\gamma_i}) \cup (\bigcup \gamma_j)$

$(\forall i, j$: $\alpha_i, \alpha_j \in body(r_i)$, $\gamma_i, \gamma_j \in body(t_i)$, $\alpha_i, \gamma_i \in V_i$, $\alpha_j, \gamma_j \notin V_i)$ *and*

(ϵ) $\forall s_i \in R^{sd}[\neg x_i]$ *either*

(ϵ_1) $\exists \beta \in body(s_i)$ s.t. $w \in S_\beta$ *or*

$(\epsilon_2) \exists t_i \in R^{sd}[x_i]$:

$(\epsilon_{2.1})$ $\forall \gamma \in body(t_i)$: $S_\gamma \in Pr(1...n)$ *and* $S_\gamma = SS_\gamma$ *and*
$(\epsilon_{2.2})$ $Stronger((\bigcup SS_{\gamma_i}) \cup (\bigcup \gamma_j), (\bigcup SS_{\beta_i}) \cup (\bigcup \beta_j), T_i) = (\bigcup SS_{\gamma_i}) \cup (\bigcup \gamma_j)$
$(\forall i, j$: $\gamma_i, \gamma_j \in body(t_i)$, $\beta_i, \beta_j \in body(s_i)$, $\gamma_i, \beta_i \in V_i$, $\alpha_j, \beta_j \notin V_i) \Rightarrow$
$S_{x_i} = S_{r_i} = SS_{r_i^{lm}} = \Sigma$

If $Ans_{x_i} = No$ derives in the first $(n+1)$ calls of $P2P\_DR_{dl}$:

$Ans_{x_i} = No \Rightarrow$
(α) $localAns_{x_i} \neq Yes$ *and*
(β) $localAns_{\neg x_i} \neq Yes$ *and*
(γ) $\exists s_i^{ldm} \in R_c(x_i)$:
(γ_1) $\forall \beta \in body(s_i^{ldm})$: $Ans_\beta = Yes$ *and*
(γ_2) $\forall r_i^{ldm} \in R_s(x_i)$ *either*
$(\gamma_{2.1})$ $\exists \alpha \in body(r_i^{ldm})$ s.t. $Ans_\alpha = No$ *or*
$(\gamma_{2.2})$ $s_i^{ldm} > r_i^{ldm}$ *or*
$(\gamma_{2.3})$ $Stronger(SS_{r_i^{ldm}}, SS_{s_i^{ldm}}, T_i) \neq SS_{r_i^{ldm}}$ *and* $r_i^{ldm} \not> s_i^{ldm} \Rightarrow$

(α) $localAns_{x_i} \neq Yes$ *and*
(β) $localAns_{\neg x_i} \neq Yes$ *and*
(γ) $\exists s_i^{ldm} \in R_c(x_i)$:
(γ_1) $\forall \beta \in body(s_i^{ldm})$: $Ans_\beta = Yes$ *and*
(γ_2) $\forall r_i^{ldm} \in R_s(x_i)$ *either*
$(\gamma_{2.1})$ $\exists \alpha \in body(r_i^{ldm})$ s.t. $Ans_\alpha = No$ *or*
$(\gamma_{2.2})$ $s_i^{ldm} > r_i^{ldm}$ *or*
$(\gamma_{2.3})$ $Stronger((\bigcup SS_{\alpha_i}) \cup (\bigcup \alpha_j), (\bigcup SS_{\beta_i}) \cup (\bigcup \beta_j), T_i) \neq (\bigcup SS_{\alpha_i}) \cup (\bigcup \alpha_j)$
$(\forall i, j$: $\alpha_i, \alpha_j \in body(r_i^{ldm})$, $\beta_i, \beta_j \in body(s_i^{ldm})$, $\alpha_i, \beta_i \in V_i$, $\alpha_j, \beta_j \notin V_i)$
and $r_i^{ldm} \not> s_i^{ldm}$
$\Rightarrow$ ((7)(8), Theorems 3 and 5)

(α) $S_{x_i} \neq \{\}$ *and*
(β) $S_{\neg x_i} \neq \{\}$ *and*
(γ) $\exists s_i \in T_\upsilon(P)$: $s_i \in R^{sd}[\neg x_i]$ *and*
(γ_1) $\forall \beta \in body(s_i)$: $S_\beta \in Pr(1...n)$ *and* $S_\beta = SS_\beta$ *and*
(γ_2) $\forall r_i \in R^{sd}[x_i]$ *either*
$(\gamma_{2.1})$ $\exists \alpha \in body(r_i)$ s.t. $S_\alpha \in Pr(1...n)$ *and* $w \in S_\alpha$ *or*
$(\gamma_{2.2})$ $s_i^{ldm} > r_i^{ldm}$ *or*
$(\gamma_{2.3})$ $Stronger((\bigcup SS_{\alpha_i}) \cup (\bigcup \alpha_j), (\bigcup SS_{\beta_i}) \cup (\bigcup \beta_j), T_i) \neq (\bigcup SS_{\alpha_i}) \cup (\bigcup \alpha_j)$
$(\forall i, j$: $\alpha_i, \alpha_j \in body(r_i)$, $\beta_i, \beta_j \in body(s_i)$, $\alpha_i, \beta_i \in V_i$, $\alpha_j, \beta_j \notin V_i)$
and $r_i \not> s_i \Rightarrow w \in S_{x_i}$

Right to Left Proof: Induction on the derivation steps in $Pr(1...n)$.

Base Case. We will prove that for a literal x_i for which $localAns_{x_i} = No \Leftrightarrow S_{x_i} \neq \{s\}$:

(9) $P(2) = S_{x_i} = \Sigma$ *and* $w \notin \Sigma \Rightarrow Ans_{x_i} = Yes$ *and* $SS_{x_i} = T$, *and*

(10) $P(1) = S_{x_i} = \Sigma$ *and* $w \in \Sigma \Rightarrow Ans_{x_i} = No$

(The supportive set of a literal cannot derive in the first step of the derivation process, unless it contains w)

(9) $P(2) = S_{x_i} = \Sigma$ *and* $w \notin \Sigma$ *and* $\Sigma \neq \{s\} \Rightarrow$
(α) $\exists r_i \in R[x_i]$: $S_{r_i} = \Sigma$ *and* $w \notin \Sigma$ *and* $\Sigma \neq \{s\}$ *and*
(β) $\nexists s_i \in R[\neg x_i]$ *and*
(γ) $\nexists t_i \neq r_i$: $t_i \in R[x_i] \Rightarrow$

(α) $\exists r_i \in R^d[x_i]$: $S_{r_i} = \Sigma$ *and* $body(r_i) = \{\}$ *and* $\Sigma = \{\}$ *and*
(β) $\nexists s_i \in R[\neg x_i]$ *and*
(γ) $\nexists t_i \neq r_i$: $t_i \in R[x_i] \Rightarrow$ (Theorem 3)

(α) $\exists r_i^d \in R_s(x_i)$: $S_{r_i^d} = \Sigma$ *and* $body(r_{i^d}) = \{\}$ *and* $\Sigma = \{\}$ *and*
(β) $\nexists s_i^{ldm} \in R_c(x_i)$ *and*
(γ) $\nexists t_i^{ldm} \neq r_i^d$: $t_i^{ldm} \in R_s(x_i) \Rightarrow$ (Theorem 3)

$Ans_{x_i} = Yes$ *and* $SS_{x_i} = \{\} = \Sigma$

(10) $P(1) = S_{x_i} = \Sigma$ *and* $w \in \Sigma \Rightarrow$
$\nexists r_i \in T_\upsilon(P)$: $r_i \in R^s[x_i] \Rightarrow$ (Theorem 3)
$\nexists r_i^{ldm} \in R_s(x_i) \Rightarrow$
$Ans_{x_i} = No$

Induction Step. Assume that for a literal x_i

(11) $\Sigma = S_{x_i} \in P(n)$ *and* $w \notin \Sigma$ *and* $\Sigma \neq \{s\} \Rightarrow$
$Ans_{x_i} = Yes$ *and* $SS_{x_i} = \Sigma$, *and*

(12) $\Sigma = S_{x_i} \in P(n)$ *and* $w \in \Sigma \Rightarrow$
$Ans_{x_i} = No$

$S_{x_i} = \Sigma \in Pr(n+1)$ *and* $w \notin \Sigma$ *and* $S_{x_i} \neq \{s\} \Rightarrow$

(α) $\exists r_i \in T_\upsilon(P)$: $S_{r_i} = \Sigma \in Pr(1...n)$ *and* $r_i \in R^{sd}[x_i]$ *and* $w \notin \Sigma$ *and* $\Sigma \neq \{s\}$ *and*
(β) $\forall s_i \in R^{sd}[\neg x_i]$: $\exists q_i \in R^{sd}[x_i]$ s.t. $S_{q_i} \in Pr(1...n)$*and*
(β_1) $w \notin S_{q_i}$ *and* $q_i > s_i \in P_i$ *or*
(β_2) $q_i > s_i \in Pr(1...n)$ *and*
(γ) $\forall t_i \in R^{sd}[x_i]$:
$S_{t_i} \in Pr(1...n)$ *and* $Stronger(S_{t_i}, S_{r_i}, T_i) \neq S_{t_i} \Rightarrow$

(α) $\exists r_i \in T_\upsilon(P)$: $S_{r_i} = \Sigma \in Pr(1...n)$ *and* $r_i \in R^{sd}[x_i]$ *and* $\Sigma \neq \{s\}$ *and*
$\forall \alpha \in body(r_i)$: $S_\alpha \in Pr(1...n)$ *and* $w \notin S_\alpha$ *and*
(β) $\forall s_i \in R^{sd}[\neg x_i]$: $\exists q_i \in R^{sd}[x_i]$ s.t. $S_{q_i} \in Pr(1...n)$*and*
$\forall \delta \in body(q_i)$: $S_\delta \in Pr(1...n)$ *and* $w \notin S_\delta$ *and*
(β_1) $q_i > s_i \in P_i$ *or*
(β_2) $Stronger(S_{q_i}, S_{s_i}, T_i) = S_{q_i}$ *and*
(γ) $\forall t_i \in R^{sd}[x_i]$:
$S_{t_i} \in Pr(1...n)$ *and* $Stronger(S_{t_i}, S_{r_i}, T_i) \neq S_{t_i} \Rightarrow$

(α) $\exists r_i \in T_\upsilon(P)$: $S_{r_i} = \Sigma \in Pr(1...n)$ *and* $r_i \in R^{sd}[x_i]$ *and* $\Sigma \neq \{s\}$ *and*
$\forall \alpha \in body(r_i)$: $S_\alpha \in Pr(1...n)$ *and* $w \notin S_\alpha$ *and*
(β) $\forall s_i \in R^{sd}[\neg x_i]$: $\exists q_i \in R^{sd}[x_i]$ s.t. $S_{q_i} \in Pr(1...n)$*and*
$\forall \delta \in body(q_i)$: $S_\delta \in Pr(1...n)$ *and* $w \notin S_\delta$ *and*
(β_1) $q_i > s_i \in P_i$ *or*
(β_2) $\exists \beta \in body(s_i)$ s.t. $S_\beta \in Pr(1...n)$ *and* $w \in S_\beta$ *or*
(β_3) $\forall \beta \in body(s_i)$: $S_\beta \in Pr(1...n)$ *and* $w \notin S_\beta$ *and*
$Stronger((\bigcup S_{\delta_i}) \cup (\bigcup \delta_j), (\bigcup S_{\beta_i}) \cup (\bigcup \beta_j), T_i) = (\bigcup S_{\delta_i}) \cup (\bigcup \delta_j)$
($\forall i, j$: $\delta_i, \delta_j \in body(q_i)$, $\beta_i, \beta_j \in body(s_i)$, $\delta_i, \beta_i \in V_i$, $\delta_j, \beta_j \notin V_i$) *and*
(γ) $\forall t_i \in R^{sd}[x_i]$: $S_{t_i} \in Pr(1...n)$ *and either*
(γ_1) $\exists \gamma$: $S_\gamma \in Pr(1...n)$ *and* $w \in S_\gamma$ *or*
(γ_2) ($\gamma_{2.1}$) $\forall \gamma \in body(t_i)$: $S_\gamma \in Pr(1...n)$ *and* $w \notin S_\gamma$ *and*
($\gamma_{2.2}$) $Stronger((\bigcup SS_{\gamma_i}) \cup (\bigcup \gamma_j), (\bigcup S_{\alpha_i}) \cup (\bigcup \alpha_j), T_i) \neq (\bigcup S_{\gamma_i}) \cup (\bigcup \gamma_j)$
($\forall i, j$: $\alpha_i, \alpha_j \in body(r_i)$, $\gamma_i, \gamma_j \in body(t_i)$, $\alpha_i, \gamma_i \in V_i$, $\alpha_j, \gamma_j \notin V_i$)
$\Rightarrow$ ((11),(12), Theorems 3,5)

(α) $\exists r_i^{ldm}$: $S_{r_i^{ldm}} = \Sigma \in Pr(1...n)$ *and* $r_i^{ldm} \in R_s(x_i)$ *and*
$\forall \alpha \in body(r_i^{ldm})$: $Ans_\alpha = Yes$ *and* $SS_\alpha = S_\alpha$ *and*
(β) $\forall s_i^{ldm} \in R_c(x_i)$: $\exists q_i^{ldm} \in R_s(x_i)$ s.t.
$\forall \delta \in body(q_i^{ldm})$: $Ans_\alpha = Yes$ *and* $SS_\alpha = S_\alpha$ *and*
(β_1) $q_i^{ldm} > s_i^{ldm} \in P_i$ *or*

(β_2) $\exists\beta \in body(s_i^{ldm})$ s.t. $Ans_\beta = No$ *or*
(β_3) $Ans_\beta = Yes$ *and* $SS_\beta = S_\beta$ *and*
$Stronger((\bigcup S_{\delta_i}) \cup (\bigcup \delta_j), (\bigcup S_{\beta_i}) \cup (\bigcup \beta_j), T_i) = (\bigcup S_{\delta_i}) \cup (\bigcup \delta_j)$
$(\forall i, j\colon \delta_i, \delta_j \in body(q_i^{ldm}),\ \beta_i, \beta_j \in body(s_i^{ldm}),\ \delta_i, \beta_i \in V_i,\ \delta_j, \beta_j \notin V_i)$ *and*
(γ) $\forall t_i^{ldm} \in R_s(x_i)$: *either*
(γ_1) $\exists\gamma \in body(t_i^{ldm})$ s.t. $Ans_\gamma = No$ *or*
(γ_2) $(\gamma_{2.1})$ $\forall\gamma \in body(t_i^{ldm})$: $Ans_\gamma = Yes$ *and* $SS_\gamma = S_\gamma$ *and*
$(\gamma_{2.2})$ $Stronger((\bigcup SS_{\gamma_i}) \cup (\bigcup \gamma_j), (\bigcup S_{\alpha_i}) \cup (\bigcup \alpha_j), T_i) \neq (\bigcup S_{\gamma_i}) \cup (\bigcup \gamma_j)$
$(\forall i, j\colon \alpha_i, \alpha_j \in body(r_i^{ldm}),\ \gamma_i, \gamma_j \in body(t_i^{ldm}),\ \alpha_i, \gamma_i \in V_i,\ \alpha_j, \gamma_j \notin V_i) \Rightarrow$

$Ans_{x_i} = Yes$ *and* $SS_{x_i} = S_{x_i} = \Sigma$

$S_{x_i} = \Sigma \in Pr(n+1)$ *and* $w \in \Sigma \Rightarrow$

(α) $\forall r_i \in R^{sd}[x_i]$: $S_{r_i} \in Pr(1...n)$ *and* $w \in S_{r_i}$ *or*
(β) $\exists s_i \in R^{sd}[\neg x_i]$: $S_{s_i} = \Sigma \in Pr(1...n)$ *and* $w \notin S_{s_i}$ *and* $\forall r_i \in R^{sd}[x_i]$: $S_{r_i} \in Pr(1...n)$ *and either*
(α_1) $w \in S_{r_i}$ *or*
(α_2) $s_i > r_i \in P_i$ *or*
(α_3) $r_i > s_i \notin P_i$ *and* $Stronger(S_{r_i}, S_{s_i}, T_i) \neq S_{r_i}$ *or*
(γ) $S_{\neg x_i} \in Pr(1...n)$ *and* $S_{\neg x_i} = \{s\} \Rightarrow$

(α) $\forall r_i \in R^{sd}[x_i]$: $\exists\alpha \in body(r_i)$: $w \in S_\alpha$ *or*
(β) $\exists s_i \in R^{sd}[\neg x_i]$: $S_{s_i} = \Sigma \in Pr(1...n)$ *and* $\forall r_i \in R^{sd}[x_i]$: $S_{r_i} \in Pr(1...n)$ *and either*
(β_1) $\exists\alpha \in body(r_i)$: $w \in S_\alpha$ *or*
(β_2) $s_i > r_i \in P_i$ *or*
(β_3) $(\beta_{3.1})$ $r_i > s_i \notin P_i$ *and*
$(\beta_{3.2})$ $\forall\alpha \in body(r_i)$: $S_\alpha in Pr(1...n)$ *and* $w \notin S_\alpha$ *and*
$(\beta_{3.3})$ $Stronger((\bigcup S_{\alpha_i}) \cup (\bigcup \alpha_j), (\bigcup S_{\beta_i}) \cup (\bigcup \beta_j), T_i) \neq (\bigcup S_{\alpha_i}) \cup (\bigcup \alpha_j)$
$(\forall i, j\colon \alpha_i, \alpha_j \in body(r_i),\ \beta_i, \beta_j \in body(s_i),\ \alpha_i, \beta_i \in V_i,\ \alpha_j, \beta_j \notin V_i)$ *or*
(γ) $S_{\neg x_i} \in Pr(1...n)$ *and* $S_{\neg x_i} = \{s\} \Rightarrow$ ((11),(12), Theorems 3,5)

(α) $\forall r_i^{ldm} \in R_s(x_i)$: $\exists\alpha \in body(r_i^{ldm})$: $Ans_\alpha = No$ *or*
(β) $\exists s_i^{ldm} \in R_c(x_i)$: $S_{s_i^{ldm}} = \Sigma$ *and* $\forall r_i^{ldm} \in R_s(x_i)$ *either*
(β_1) $\exists\alpha \in body(r_i^{ldm})$: $Ans_\alpha = No$ *or*
(β_2) $s_i > r_i \in P_i$ *or*
(β_3) $(\beta_{3.1})$ $r_i > s_i \notin P_i$ *and*
$(\beta_{3.2})$ $\forall\alpha \in body(r_i^{ldm})$: $Ans_\alpha = Yes$ *and* $S_\alpha = SS_\alpha$ *and*
$(\beta_{3.3})$ $Stronger((\bigcup S_{\alpha_i}) \cup (\bigcup \alpha_j), (\bigcup S_{\beta_i}) \cup (\bigcup \beta_j), T_i) \neq (\bigcup S_{\alpha_i}) \cup (\bigcup \alpha_j)$
$(\forall i, j\colon \alpha_i, \alpha_j \in body(r_i^{ldm}),\ \beta_i, \beta_j \in body(s_i^{ldm}),\ \alpha_i, \beta_i \in V_i,\ \alpha_j, \beta_j \notin V_i)$ *or*

(γ) $localAns_{\neg x_i} = Yes \Rightarrow$

$Ans_{x_i} = No$

Following Theorem 6, it is straightforward to prove that Lemma 7 holds here too, with some minor modifications:

Lemma 7: *For any literal x_i for which, $localAns_{x_i} = No$ and $localAns_{\neg x_i} = No$ and for any two local or mapping rules $r_i^{ldm} \in R_s(x_i)$, $s_i^{ldm} \in R_c(x_i)$*
(for which there is no priority relation in P_i)
and their corresponding rules $r_i, s_i \in T_\upsilon(P)$:

$$Stronger(SS_{r_i^{lm}}, SS_{s_i^{lm}}, T_i) = SS_{r_i^{lm}} \Leftrightarrow r_i > s_i \in Pr(1...n)$$

Theorem 8, which states the equivalency between the local reasoning process (*local_alg*) of the distributed algorithm, and the derivation of definite proofs in the unified defeasible theory $T_\upsilon(P)$ holds as it is in this version as well. The proof for it is exactly the same, with the one that we presented for the first version of $P2P\_DR$.

Theorem 8: $T_\upsilon(P) \vdash +\Delta x_i$ *is equivalent to* $localAns_{x_i} = Yes$ *and*
$T_\upsilon(P) \vdash -\Delta x_i$ *is equivalent to* $localAns_{x_i} = No$

Theorem 9, which describes the correlation between the answers returned by the distributed reasoning algorithm, and the defeasible proofs returned by $T_\upsilon(P)$, holds also for this version. Below, we give the theorem and its proof, which is slightly different from the one that we presented in the previous chapter.

Theorem 9: $T_\upsilon(P) \vdash +\partial x_i$ *is equivalent to* $Ans_{x_i} = Yes$ *and*
$T_\upsilon(P) \vdash -\partial x_i$ *is equivalent to* $Ans_{x_i} = No$

We will prove the *left-to-right part* of this theorem using Induction on the number of proof derivation steps in $T_\upsilon(P)$. A defeasible proof about a literal q cannot derive in one step, as even if there is only one supportive defeasible rule with empty body, in order to prove $+\partial q$, we should priorly derive $-\Delta\neg q$. So the base of the induction will be the first two steps of the derivation process.

Base Case. We will prove that:

(13) $P(2) = +\partial x_i \Rightarrow Ans_{x_i} = Yes$, and

(14) $P(2) = -\partial x_i \Rightarrow Ans_{x_i} = No$

(13) $P(2) = +\partial x_i \Rightarrow$
(α) $P(1) = +\Delta x_i$ *or*
(β) (β_1) $\exists r \in R^{sd}[x_i]$: $body(r) = \{\}$ *and*
(β_2) $P(1) = -\Delta \neg x_i$ *and*
(β_3) $\nexists s \in R[\neg x_i] \Rightarrow$ (Theorems 3 and 8)

(α) $localAns_{x_i} = Yes$ *or*
(β) (β_1) $\exists r_i^{ldm} \in R_s(x_i)$: $body(r_i^{ldm}) = \{\}$ *and*
(β_2) $localAns_{\neg x_i} = No$ *and*
(β_3) $\nexists s_i^{ldm} \in R_c(x_i) \Rightarrow$

(α) $Ans_{x_i} = Yes$ *or*
(β) (β_1) $\exists r_i^{ldm} \in R_s(x_i)$: $body(r_i^{ldm}) = \{\}$ *and*
(β_2) $localAns_{\neg x_i} = No$ *and*
(β_3) $CR_{x_i} = \{\} \Rightarrow$
$Ans_{x_i} = Yes$

(14) $P(2) = -\partial x_i \Rightarrow$
(α) $P(1) = -\Delta x_i$ *and*
(β) (β_1) $\nexists r \in R^{sd}[x_i]$ *or*
(β_2) $P(1) = +\Delta \neg x_i \Rightarrow$ (Theorems 3 and 8)

(α) $localAns_{x_i} = No$ *and*
(β) (β_1) $\nexists r_i^{ldm} \in R_s(x_i)$ *or*
(β_2) $localAns_{\neg x_i} = Yes \Rightarrow$

(α) $localAns_{x_i} = No$ *and*
(β) (β_1) $SR_{x_i} = \{\}$ *or*
(β_2) $localAns_{\neg x_i} = Yes \Rightarrow$
$Ans_{x_i} = No$

Induction Step. Assume that

(15) $+\partial x_i \in P(1...n) \Rightarrow Ans_{x_i} = Yes$, and

(16) $-\partial x_i \in P(1...n) \Rightarrow Ans_{x_i} = No$

For $i = n+1$

$+\partial x_i \in P(1...n+1) \Rightarrow$
- (α) $+\Delta x_i \in P(1...n)$ *or*
- (β) (β_1) $\exists r \in R^{sd}[x_i]$ s.t. $\forall \alpha \in body(r)$: $+\partial\alpha \in P(1...n)$ *and*
  - (β_2) $-\Delta\neg x_i \in P(1...n)$ *and*
  - (β_3) $\forall s \in R^{sd}[\neg x_i]$
    - $(\beta_{3.1})$ $\exists \beta \in body(s)$: $-\partial\beta \in P(1...n)$ *or*
    - $(\beta_{3.2})$ $\exists t \in R^{sd}[x_i]$:
      - $\forall \gamma \in body(t)$: $+\partial\gamma \in P(1...n)$ *and* $t > s \Rightarrow$ ((15),(16), Theorems 3,8, Lemma 7)

- (α) $localAns_{x_i} = Yes$ *or*
- (β) (β_1) $\exists r_i^{ldm} \in R_s(x_i)$ s.t. $\forall \alpha \in body(r)$: $Ans_\alpha = Yes$ *and*
  - (β_2) $localAns_{\neg x_i} = No$ *and*
  - (β_3) $\forall s_i^{ldm} \in R_c(x_i)$
    - $(\beta_{3.1})$ $\exists \beta \in body(s_i^{ldm})$: $Ans_\beta = No$ *or*
    - $(\beta_{3.2})$ $\exists t_i^{ldm} \in R_s(x_i)$:
      - $\forall \gamma \in body(t_i^{ldm})$: $Ans_\gamma = Yes$ *and*
      - $t_i^{ldm} > s_i^{ldm} \in P_i$ *or*
      - $s_i^{ldm} > t_i^{ldm} \notin P_i$ *and* $Stronger(SS_{t_i^{ldm}}, SS_{s_i^{ldm}}, T_i) = SS_{t_i^{ldm}}) \Rightarrow$

$Ans_{x_i} = Yes$

For negative provability:

$-\partial x_i \in P(1...n+1) \Rightarrow$
- (α) $-\Delta x_i \in P(1...n)$ *and*
- (β) (β_1) $\forall r \in R^{sd}[x_i]$: $\exists \alpha \in body(r)$: $-\partial\alpha \in P(1...n)$ *or*
  - (β_2) $+\Delta\neg x_i \in P(1...n)$ *or*
  - (β_3) $\exists s \in R^{sd}[\neg x_i]$ s.t.
    - $(\beta_{3.1})$ $\forall \beta \in body(s)$: $+\partial\beta \in P(1...n)$ *and*
    - $(\beta_{3.2})$ $\forall t \in R^{sd}[x_i]$:
      - $\exists \gamma \in body(t)$: $-\partial\gamma \in P(1...n)$ *or* $t \not> s \Rightarrow$ ((15),(16), Theorems 3,8, Lemma 7)

(α) $localAns_{x_i} = No$ *and*
(β) (β_1) $\forall r_i^{ldm} \in R_s(x_i)$: $\exists \alpha \in body(r_i^{ldm})$: $Ans_\alpha = No$ *or*
(β_2) $localAns_{\neg x_i} = Yes$ *or*
(β_3) $\exists s_i^{ldm} \in R_c(x_i)$:
$(\beta_{3.1})$ $\forall \beta \in body(s_i^{ldm})$: $Ans_\beta = Yes$ *and*
$(\beta_{3.2})$ $\forall t_i^{ldm} \in R_s(x_i)$:
$\exists \gamma \in body(t_i^{ldm})$: $Ans_\gamma = No$ *or*
$s_i^{ldm} > t_i^{ldm} \in P_i$ *or*
$t_i^{ldm} > s_i^{ldm} \notin P_i$ *and* $Stronger(SS_{t_i^{ldm}}, SS_{s_i^{ldm}}, T_i) \neq SS_{t_i^{ldm}} \Rightarrow$

$Ans_{x_i} = No$

We will now prove the *right-to-left part* of Theorem 9 using Induction on the number of calls of $P2P\_DR_{dl}$ that are required to compute an answer for a literal x_i.

Base Case. We will prove that:

(17) If $Ans_{x_i} = Yes$ derives at the first call of $P2P\_DR_{dl}$ then $T_\upsilon(P) \vdash +\partial x_i$, and

(18) If $Ans_{x_i} = No$ derives at the first call of $P2P\_DR_{dl}$ then $T_\upsilon(P) \vdash -\partial x_i$

(17) $Ans_{x_i} = Yes$ derives at the first call of $P2P\_DR \Rightarrow$

(α) $localAns_{x_i} = Yes$ *or*
(β) (β_1) $localAns_{\neg x_i} = No$ *and*
(β_2) $\forall r_i^{dm} \in R_s(x_i)$: $body(r_i^{dm}) = \{\}$ *and*
(β_3) $\forall s_i^{dm} \in R_c(x_i)$: $body(s_i^{dm}) = \{\}$ *and* $\exists r_i^{dm} \in R_s(x_i)$: $t_i^{dm} > s_i^{dm} \in P_i \Rightarrow$ (Theorems 3,8)

(α) $T_\upsilon(P) \vdash +\Delta x_i$ *or*
(β) (β_1) $T_\upsilon(P) \vdash -\Delta \neg x_i$ *and*
(β_2) $\forall r_i \in T_\upsilon(P)$ s.t. $r_i \in R^d[x_i]$: $body(r_i) = \{\}$ *and*
(β_3) $\forall s_i \in T_\upsilon(P)$ s.t. $s_i \in R^d[\neg x_i]$: $body(s_i) = \{\}$ *and* $\exists t_i \in R^d[x_i]$: $t_i > s_i \Rightarrow$

$T_\upsilon(P) \vdash +\partial x_i$

(18) $Ans_{x_i} = No$ derives at the first call of $P2P\_DR \Rightarrow$

(α) $localAns_{\neg x_i} = Yes$ *or*
(β) (β_1) $localAns_{x_i} = No$ *and*
(β_2) $\forall r_i^{dm} \in R_s(x_i)$: $body(r_i^{dm}) = \{\}$ *and*
(β_3) $\forall s_i^{dm} \in R_c(x_i)$: $body(s_i^{dm}) = \{\}$ *and*
(β_4) $\exists q_i^{dm} \in R_c(x_i)$: $\forall t_i^{dm} \in R_s(x_i$: $t_i^{dm} > s_i^{dm} \notin P_i \Rightarrow$ (Theorems 3,8)

(α) $T_\upsilon(P) \vdash +\Delta\neg x_i$ *or*
(β) (β_1) $T_\upsilon(P) \vdash -\Delta\neg x_i$ *and*
(β_2) $\forall r_i \in T_\upsilon(P)$ s.t. $r_i \in R^d[x_i]$: $body(r_i) = \{\}$ *and*
(β_3) $\forall s_i \in T_\upsilon(P)$ s.t. $s_i \in R^d[\neg x_i]$: $body(s_i) = \{\}$ *and*
(β_4) $\exists q_i \in R^d[\neg x_i]$: $\forall r_i \in R^d[x_i]$: $t_i \not> s_i \Rightarrow$

$T_\upsilon(P) \vdash -\partial x_i$

Induction Step. Assume that

(19) $Ans_{x_i} = Yes$ derives in the first n calls of $P2P\_DR_{dl} \Rightarrow$
$T_\upsilon(P) \vdash +\partial x_i$, and

(20) $Ans_{x_i} = No$ derives in the first n calls of $P2P\_DR_{dl} \Rightarrow$
$T_\upsilon(P) \vdash -\partial x_i$

If $Ans_{x_i} = Yes$ derives in $(n+1)$ calls of $P2P\_DR_{dl} \Rightarrow$

(α) $localAns_{x_i} \neq Yes$ *and*
(β) $localAns_{\neg x_i} \neq Yes$ *and*
(γ) $\exists r_i^{ldm} \in R_s(x_i)$:
$\forall \alpha \in body(r_i^{ldm})$: $Ans_\alpha = Yes$ (in n calls) *and*
(δ) $\forall s_i^{ldm} \in R_c(x_i)$ *either*
(δ_1) $\exists \beta \in body(s_i^{ldm})$ s.t. $Ans_\beta = No$ (in n calls) *or*
(δ_2) $\exists t_i^{ldm} \in R_s(x_i)$:
($\delta_{2.1}$) $\forall \gamma \in body(t_i^{ldm})$: $Ans_\gamma = Yes$ (in n calls) *or*
($\delta_{2.2}$) $t_i^{ldm} > s_i^{lm} \in P_i$ *or* $s_i^{ldm} > t_i^{ldm} \notin P_i$ *and* $Stronger(SS_{t_i^{lm}}, SS_{s_i^{lm}}, T_i) = SS_{t_i^{lm}}$
$\Rightarrow$ ((19)(20), Theorems 3,8, Lemma 7)

(α) $-\Delta x_i$ *and*
(β) $-\Delta\neg x_i$ *and*

(γ) $\exists r \in T_\upsilon(P)$: $r \in R^{sd}[x_i]$ *and* $\forall \alpha \in body(r)$: $+\partial\alpha$ *and*
$\delta)$ $\forall s \in R^{sd}[x_i]$ *either*
(δ_1) $\exists \beta \in body(s)$ s.t. $-\partial\beta$ *or*
(δ_2) $\exists t \in T_\upsilon(P)$: $t \in R^{sd}[x_i]$ *and*
$(\delta_{2.1})$ $\forall \gamma \in body(t)$: $+\partial\gamma$ *or*
$(\delta_{2.2})$ $t > s \in T_\upsilon(P) \Rightarrow$

$T_\upsilon(P) \vdash +\partial x_i$

If $Ans_{x_i} = No$ derives in $(n+1)$ calls of $P2P\_DR \Rightarrow$

(α) $localAns_{x_i} \neq Yes$ *and*
(β) $localAns_{\neg x_i} \neq Yes$ *and either*
(γ) (γ_1) $\forall r_i^{ldm} \in R_s(x_i)$: $\exists \alpha \in body(r_i^{ldm})$ s.t. $Ans_\alpha = No$ (in at most n calls) *or*
(γ_2) $\exists s_i^{ldm} \in R_c(x_i)$:
$(\gamma_{2.1})$ $\forall \beta \in body(s_i^{ldm})$: $Ans_\beta = Yes$ (in n calls) *and*
$(\gamma_{2.2})$ $\forall r_i^{ldm} \in R_s(x_i)$: *either*
$(\gamma_{2.2.1})$ $\exists \alpha \in body(r_i^{ldm})$ s.t. $Ans_\alpha = No$ *or*
$(\gamma_{2.2.2})$ $s_i^{ldm} > r_i^{ldm} \in P_i$ *or*
$r_i^{ldm} > s_i^{ldm} \notin P_i$ *and* $Stronger(SS_{r_i^{ldm}}, SS_{s_i^{ldm}}, T_i) \neq SS_{r_i^{ldm}}$
$\Rightarrow$ ((19)(20), Theorems 3,8, Lemma 7)

(α) $-\Delta x_i$ *and*
(β) $-\Delta \neg x_i$ *and*
(γ) (γ_1) $\forall r \in R^{sd}[x_i]$: $\exists \alpha \in body(r)$ s.t. $-\partial\alpha$ *or*
(γ_2) $\exists s \in T_\upsilon(P)$: $s \in R^{sd}[\neg x_i]$ *and*
$(\gamma_{2.1})$ $\forall \beta \in body(s)$: $+\partial\beta$ *and*
$(\gamma_{2.2})$ $\forall r \in R^{sd}[x_i]$: *either*
$(\gamma_{2.2.1})$ $\exists \alpha \in body(r)$ s.t. $-\partial\alpha$ *or*
$(\gamma_{2.2.2})$ $r_i^{ldm} > s_i^{lm} \notin T_\upsilon(P) \Rightarrow$
$T_\upsilon(P) \vdash -\partial x_i$

5 The 2nd Approach

The 1st approach, in both versions that we described, each queried peer is required to return a single positive/negative answer for the queried literal. When a conflict arises, a peer uses the trust information of the peers it queried, to evaluate the quality of the answers that they returned. Each answer is indirectly assigned with the trust value of the peer that returned this answer.

In this second approach, we attempt to associate the quality of the answer not only with the trust level of the queried peer, but also with the confidence of the queried peer on the answer it returns. Specifically, we define two levels of quality for each positive answer; (a) the *strict answers*, which derive from the local rules of the queried peer theory; and $(b)$the *weak* answers, which are based on the mappings that the queried literal has established with other system nodes. In the case that the local theories are augmented with defeasible rules, the answers that are based on local defeasible rules fall into the second category. Below, we present $P2P\_DR_2$, a version of the $P2P\_DR$ distributed algorithm that supports the features that we described.

5.1 The $P2P\_DR_2$ Algorithm

The only differences with the $P2P\_DR$ algorithm are two:

- A peer (say P_i) may return three different answers for a queried literal (say x_i). These are: (a) Yes_s, in case a positive answer for x_i derives from *local_alg* in P_i, (b), (β) Yes_w, in case P_i computes a positive answer for x_i, which does not derive from *local_alg*, (γ) No, in any other case.
- Comparing the strength of two supportive sets is not only based on the trust value of the peers, which have defined the literals contained in these sets, but also on the level of answer for these literals. Specifically, a *strict* answer for one literal is considered stronger than a *weak* answer for another literal, independently of the trust level of the peers that have defined these two literals. Comparing the strength of two literals with the same level of proof is again entirely based on the trust level of the peers that define these literals.

For example, assume that in P_i there is one supportive mapping rule for x_i, m_1: $a_k \Rightarrow x_i$, and one mapping rule that contradicts x_i, m_2: $b_l \Rightarrow \neg x_i$, and P_l precedes P_k in T_i. Assume also, that a_k is proved based on the local rules of P_k, whereas P_l computes a positive answer for b_l using its mappings. The first version of the algorithm, $P2P\_DR$, would compute a *negative* answer for x_i, as it would compute $SS_{x_i} = \{a_k\}$, and $CS_{x_i} = \{b_l\}$, and a_k is weaker than b_l, based on T_i. On the other hand, $P2P\_DR_2$ will take into account that P_k provides a *strict* positive answer for a_k, while P_l provides a *weak* positive answer for b_l, and will eventually return a *positive* answer for x_i, as it will not take into account the trust level of P_k and P_l.

The only lines of the code of $P2P\_DR$ that we have to modify to support the three levels of answer are:

Line 5: $Ans_{x_i} \leftarrow str_{x_i}$

Lines 27-28: **else if** $Ans_{b_t} \neq No$ *and* $b_t \notin V_i$ **then** $SS_{r_i} \leftarrow SS_{r_i} \cup Ans_{b_t}$

Lines 51-52: **else if** $Ans_{b_t} \neq No$ *and* $b_t \notin V_i$ **then** $SS_{r_i} \leftarrow SS_{r_i} \cup Ans_{b_t}$

Line 63: return $Ans_{x_i} = weak_{x_i}$ and SS_{x_i} and terminate

Line 66: return $Ans_{x_i} = weak_{x_i}$ and SS_{x_i} and terminate

The *Stronger* function is also modified as follows:

Stronger$_2(S, C, T)$

if $\exists \alpha$: $Ans_a = weak_a \in S$ **then**
  $a^w \leftarrow a_k | Ans_{a_k} = weak_{a_k} \in S$ *and for all* $a_i | Ans_{a_i} = weak_{a_i} \in S$: P_k does not precede P_i in T
else
  $a^w \leftarrow a_k | Ans_{a_k} \in S$ *and for all* $a_i | Ans_{a_i} \in S$: P_k does not precede P_i in T
end if
if $\exists b$: $Ans_b = weak_b \in C$ **then**
  $b^w \leftarrow b_l | Ans_{b_l} = weak_{b_l} \in C$ *and for all* $b_j | Ans_{b_j} = weak_{b_j} \in C$: P_l does not precede P_j in T
else
  $b^w \leftarrow b_l | Ans_{b_l} \in C$ *and for all* $b_j | Ans_{b_j} \in C$: P_l does not precede P_j in T
end if
if $Ans_{a^w} = str_{a^w}$ *and* $Ans_{b^w} = weak_{b^w}$ **then**

    Stronger ← *S*
else if $Ans_{a^w} = weak_{a^w}$ *and* $Ans_{b^w} = str_{b^w}$ **then**
    Stronger ← *C*
else
    if P_k precedes P_l in T **then**
        Stronger ← *S*
    else if P_l precedes P_k in T **then**
        Stronger ← *C*
    else
        Stronger ← *None*
    end if
end if

5.2 Properties of P2P_DR$_2$

In the same way with *P2P_DR*, it is easy to prove the following properties for $P2P\_DR_2$:

- $P2P\_DR_2$ *always terminates.*
- *The total number of messages that need to be exchanged between the system nodes for the computation of a single query with regard to the total number of system nodes is in the worst case* $O(n^2)$ (using the same optimizations that we described for the case of *P2P_DR*).
- *The computational complexity of* $P2P\_DR_2$ *for the computation of a single query on one node is in the worst case* $O(n^2 \times n_l^2 \times n_r)$ (where n is the number of system nodes, n_l is the number of literals a node may define, and n_r is the number of rules a node may define.

The *INC_Q* and *OUT_Q* structures that are part of the optimizations that we described have to be slightly modified for the needs of $P2P\_DR_2$. Specifically, for each queried literal x_i, we can have four different values: (*a*) str_{x_i}; (*b*) $weak_{x_i}$; (*c*) *No*; and (*d*) *undetermined*. For the first three cases, the algorithm retrieves the stored answer. In the latter case, the algorithm call is suspended, until the computation of the answer for x_i is completed by another algorithm call that is still pending.

5.3 Equivalent Defeasible Theory

The steps that are required to build an equivalent defeasible theory from the unification of the distributed peer theories for the second version of the distributed reasoning algorithm, $P2P\_DR_2$, are similar with those that we described for the case of $P2P\_DR$. In fact, we only have to modify the *Priorities* procedure that adds priorities between conflicting rules in the unified theory. The differences between the *Priorities* procedure and the modified version, $Priorities_2$ are:

- The derivation of the Supportive Set of a rule r_i in the $(i+1)_{th}$ step of the derivation process Pr is modified to:

  If $Pr(i+1) = S_{r_i}$ *then either*
  (α) $S_{r_i} = (\bigcup S_{a_i}) \cup (\bigcup str_{a_j}) \cup (\bigcup weak_{a_k})$, *and*
  $\forall a_i$: $a_i \in V_i$, $a_i \in body(r_i)$, $S_{a_i} \in Pr(1...i)$ *and*
  $\forall a_j$: $a_j \notin V_i$, $a_j \in body(r_i)$, $S_{a_j} \in Pr(1...i)$, $S_{a_j} = \{\}$ *and*
  $\forall a_k$: $a_k \notin V_i$, $a_k \in body(r_i)$, $S_{a_k} \in Pr(1...i)$, $S_{a_j} \neq \{\}$, $w \notin S_{a_k}$ *or*
  (β) $S_{r_i} = w$, *and*
  $\exists a_j$, s.t. $a_j \notin V_i$, $a_j \in body(r_i)$, $S_{a_j} \in Pr(1...i)$, $w \in S_{a_j}$

- In the derivation of a priority relation or of the Supportive Set of a literal, instead of the *Stronger* function, we use its modified version, $Stronger_2$

Theorems 3,4 and 5 hold for $P2P\_DR_2$ and the proofs for these theorems derive in exactly the same way with the proofs that we presented for the case of the first version $P2P\_DR$. Using these three theorems, we can derive Theorem 6 in a very similar way with the one that we presented for the case of $P2P\_DR$. Lemma 7, and Theorems 8 and 9 derive also in exactly the same way with the case of $P2P\_DR$, following Theorems 3-6.

6 The 3d Approach

The 2nd version of the distributed algorithm, which we presented in the previous section, extends the first version, $P2P\_DR$, by supporting two levels for the positive answers, based on whether these answers derive from the peer's local theory or from its mappings. In this section, we describe a more extended version, in which a peer does not return a single positive/negative answer, but it augments it with its *supportive set*; namely, the foreign literals that it has to prove to reach to a true/false truth value. This set of foreign literals may not only contain literals that are involved in the local peer's mappings. For example, consider the case that P_1 is queried about literal x_1. If in order to compute an answer for x_1, P_1 has to query P_2 about x_2, and in order P_2 to be able to find the truth value of x_2, it has to query P_3 about x_3, which is locally proved in P_3, the answer returned by P_1 will contain both x_2 and x_3.

6.1 The P2P_DR$_3$ Algorithm

The steps of the third version of the algorithm, $P2P\_DR_3$, differ from the original version only in the process of building the supportive/conflicting set of a literal. In this version, the supportive set of a literal, say x_i, contains all the foreign literals that all the recursive calls have to prove in order to be able to derive a positive answer (in the absence of any contradictions). Considering that some of these algorithm calls may be executed by different peers than P_i, this set may contain literals that are not involved in P_i's mappings (but they are involved in mappings defined by other peers). If there are more than one ways to support x_i, $P2P\_DR_3$ builds the supportive sets of all the supportive rules, and keeps the one which is the *strongest* based on the trust level order of P_i, T_i. The algorithm uses the same trust information to compare the supportive set of x_i, SS_{x_i}, with the conflicting set, CS_{x_i}, to reach to the final answer.

To support these new features $P2P\_DR$ algorithm is modified as follows:

Lines 27-28: **else if** $Ans_{b_t} = Yes \ and \ b_t \notin V_i$ **then** $SS_{r_i} \leftarrow SS_{r_i} \cup SS_{b_t} \cup b_t$

Lines 51-52: **else if** $Ans_{b_t} = Yes \ and \ b_t \notin V_i$ **then** $SS_{r_i} \leftarrow SS_{r_i} \cup SS_{b_t} \cup b_t$

The *local_alg* algorithm and the *Stronger* function remain unchanged.

To clarify the difference between the two versions $P2P\_DR$ and $P2P\_DR_3$ consider the following example: A peer, say P_1, is queried about x_1 by one of its acquaintances. Assume that x_1 is supported by one mapping rule; m_{11}: $x_2 \rightarrow x_1$, and is contradicted by one mapping rule; m_{12}: $x_3 \rightarrow \neg x_1$. Assume also that in P_2 there is one rule that supports x_2; m_{21}: $x_4 \rightarrow x_2$ and no rule that contradicts it, and that x_4 is locally proved in P_4. Assume that in P_3 there is one rule that supports x_3; m_{31}: $x_5 \rightarrow x_3$ and no rule that contradicts it, and that x_5 is locally proved in P_5. Finally, assume that P_1 has defined its trust level order as follows; $T_1 = [P_3, P_2, P_4, P_5]$. The first version, $P2P\_DR$ would compute $SS_{x_1} = \{x_2\}$ and $CS_{x_1} = \{x_3\}$ and would return $Ans_{x_1} = No$, as P_3 precedes P_2 in T_1. On the other hand, the new version, $P2P\_DR_3$, would compute $SS_{x_1} = \{x_2, x_4\}$ and $CS_{x_1} = \{x_3, x_5\}$ and would return $Ans_{x_1} = Yes$, as x_4 is the weakest element of SS_{x_1} and x_5 is the weakest element of CS_{x_1}, and P_4 precedes P_5 in T_1.

6.2 Properties of P2P_DR$_3$

Theorems 1 and 2 (regarding termination and total number of messages) hold also for this version, $P2P\_DR_3$ (the proof is exactly the same). The computational complexity of this version on a single node is also in the worst case the same. The worst case is, however, different in the two cases. In general, for the case of $P2P\_DR$, the computational complexity is $O(n_{ACQ}^2 \times n_l^2 \times n_r)$, where n_{ACQ} is the number of acquaintances a peer may have, n_l is the number of literals a peer may define, and n_r is the number of rules a peer may define. The worst case is that *all peers have defined mappings that involve all literals from all system nodes*. In that case $n_{ACQ} = n$, where n is the number of system nodes, and the complexity is $O(n^2 \times n_l^2 \times n_r)$. For the case of $P2P\_DR_3$, the worst case is that *computing the truth value of every literal involves computing the truth value of all literals from all system nodes*. In that case the complexity is also $O(n^2 \times n_l^2 \times n_r)$. However, the requirement for the worst case in this version is rather more realistic than in the case of $P2P\_DR$.

6.3 Equivalent Defeasible Theory

Building an equivalent defeasible theory from the distributed local peer theories for the case of $P2P\_DR_3$ is feasible following the same procedure with the one that we described for the case of $P2P\_DR$. The only modification we have to make is in the procedure that adds priorities between conflicting rules in the unified theory. The difference between the *Priorities* procedure and the modified version, $Priorities_3$ is:

- The derivation of the Supportive Set of a rule r_i in the $(i+1)_{th}$ step of the derivation process Pr is modified to:

  If $Pr(i+1) = S_{r_i}$ *then*
  $S_{r_i} = (\bigcup S_{a_i}) \cup (\bigcup a_j) \cup (\bigcup SS_{a_j})$, *and*
  $\forall a_i$: $a_i \in V_i$, $a_i \in body(r_i)$, $S_{a_i} \in Pr(1...i)$ *and*
  $\forall a_j$: $a_j \notin V_i$, $a_j \in body(r_i)$, $S_{a_j} \in Pr(1...i)$

The proofs for Theorems 3-9 are very similar with the case of $P2P\_DR$. In fact, for Theorems 3-5, Lemma 7 and Theorems 8-9 the proofs are exactly the same.

7 The 4th Approach

The main feature of $P2P\_DR_3$ is that along with the truth value of the queried literal, a peer also returns its Supportive Set. This set describes the *most trusted* way to reach to the final answer. However, trust is subjective. The *most trusted* between two or more different ways will be different if we use the trust level orders of two different peers.

In this section, we describe another approach that addresses this issue; when a peer is queried about one of its local literals, it returns its truth value along with its Supportive Set, which in this case contains all the different ways that can be applied to support this literal. In the new version of the algorithm, $P2P\_DR_4$, the Supportive Set of a literal is actually a set of the Supportive Sets of all the rules that can be applied to support this literal. The reason for retaining the supportive sets of all supportive rules is that, although the queried peer (say P_j) may regard $SS_{r_{j1}}$ stronger (*more trusted*) than $SS_{r_{j2}}$ based on its trust level order, T_j, (where r_{i1}, r_{i2} are two supportive rules for the queried literal, x_j), the peer that issued the query, say P_i may have a different opinion based on T_i.

7.1 The P2P_DR$_4$ Algorithm

$P2P\_DR_4$ follows the four main steps of the original version, $P2P\_DR$, with two modifications in the process of building the supportive sets, and in the process of comparing two conflicting sets. In this version, whenever the algorithm computes a positive truth value for the literals that lie in the body of a supportive rule, it augments the Supportive Set of the queried literal, with the Supportive Set of this rule. It also does the same thing with the Conflicting Sets. To compare two conflicting sets, say SS_{x_i} and CS_{x_i}, it actually compares the *strongest* mapping sets of SS_{x_i} and CS_{x_i} using T_i and the *Stronger* function.

P2P_DR$(x_i, P_0, P_i, SS_{x_i}, CS_{x_i}, Hist_{x_i}, Ans_{x_i}, T_i)$

```
if ∃r_i^l ∈ R_s(x_i) then
    localHist_{x_i} ← [x_i]
    run local_alg(x_i, localHist_{x_i}, localAns_{x_i})
    if localAns_{x_i} = Yes then
        Ans_{x_i} ← localAns_{x_i}
```

```
    terminate
  end if
end if
if ∃r_i^l ∈ R_c(x_i) then
  localHist_{x_i} ← [x_i]
  run local_alg(¬x_i, localHist_{x_i}, localAns_{¬x_i})
  if localAns_{¬x_i} = Yes then
    Ans_{x_i} ← ¬localAns_{¬x_i}
    terminate
  end if
end if
for all r_i^{lm} ∈ R_s(x_i) do
  SS_{r_i} ← {}
  for all b_t ∈ body(r_i^{lm}) do
    if b_t ∈ Hist_{x_i} then
      stop and check the next rule
    else
      Hist_{b_t} ← Hist_{x_i} ∪ b_t
      run P2P_DR(b_t, P_i, P_t, SS_{b_t}, CS_{b_t}, Hist_{b_t}, Ans_{b_t}, T_t)
      if Ans_{b_t} = No then
        stop and check the next rule
      else if Ans_{b_t} = Yes and b_t ∉ V_i then
        SS_{r_i} ← SS_{r_i} × (SS_{b_t} × {b_t}) (× stands for Cartesian Product)
      else
        SS_{r_i} ← SS_{r_i} × SS_{b_t}
      end if
    end if
  end for
  SS_{x_i} ← SS_{x_i} ∪ SS_{r_i}
end for
if SS_{x_i} = {} then
  return Ans_{x_i} = No and terminate
end if
for all r_i^{lm} ∈ R_c(x_i) do
  SS_{r_i} ← {}
  for all b_t ∈ body(r_i^{lm}) do
    if b_t ∈ Hist_{x_i} then
      stop and check the next rule
    else
      Hist_{b_t} ← Hist_{x_i} ∪ b_t
```

run $P2P\_DR(b_t, P_i, P_t, SS_{b_t}, CS_{b_t}, Hist_{b_t}, Ans_{b_t}, T_t)$
if $Ans_{b_t} = No$ **then**
stop and check the next rule
else if $Ans_{b_t} = Yes\ and\ b_t \notin V_i$ **then**
$SS_{r_i} \leftarrow SS_{r_i} \times (SS_{b_t} \times \{b_t\})$
else
$SS_{r_i} \leftarrow SS_{r_i} \times SS_{b_t}$
end if
end if
end for
$CS_{x_i} \leftarrow CS_{x_i} \cup SS_{r_i}$
end for
if $CS_{x_i} = \{\}$ **then**
return $Ans_{x_i} = Yes$ and SS_{x_i} and terminate
end if
$SS^{str}_{x_i} \leftarrow SS^{m}_{x_i} \in SS_{x_i}$ s.t.
for all $SS^{j}_{x_i} \in SS_{x_i}$: $Stronger(SS^{m}_{x_i}, SS^{j}_{x_i}, T_i) \neq SS^{j}_{x_i}$
$CS^{str}_{x_i} \leftarrow CS^{m}_{x_i} \in CS_{x_i}$ s.t.
for all $CS^{j}_{x_i} \in CS_{x_i}$: $Stronger(CS^{m}_{x_i}, CS^{j}_{x_i}, T_i) \neq CS^{j}_{x_i}$
if $Stronger(SS^{str}_{x_i}, CS^{str}_{x_i}, T_i) = SS^{str}_{x_i}$ **then**
return $Ans_{x_i} = Yes$ and SS_{x_i}
else
return $Ans_{x_i} = No$
end if

The *local_alg* algorithm and the *Stronger* function remain unchanged. To clarify the difference between the two versions $P2P\_DR_3$ and $P2P\_DR_4$ consider the following example: A peer, say P_1, is queried about x_1 by one of its acquaintances. Assume that x_1 is supported by one mapping rule; m_{11}: $x_2 \rightarrow x_1$, and is contradicted by one mapping rule; m_{12}: $x_3 \rightarrow \neg x_1$. Assume also that in P_2 there are two rules that support x_2; m_{21}: $x_4 \rightarrow x_2$ and m_{22}: $x_5 \rightarrow x_2$ and no rule that contradicts it, and that x_4 is locally proved in P_4 and x_5 is locally proved in P_5. Assume that in P_3, x_3 derives from the local theory. Finally, assume that P_1 has defined its trust level order as follows; $T_1 = [P_2, P_4, P_3, P_5]$ and P_2 has defined $T_2 = [P_1, P_5, P_4]$. $P2P\_DR_3$, would compute $SS_{x_1} = \{x_2, x_5\}$ (as P_2 would return $SS_{x_2} = \{x_5\}$) and $CS_{x_1} = \{x_3\}$ and would return $Ans_{x_1} = No$ (as P_3 precedes P_5 in T_1). In the case of $P2P\_DR_4$, P_2 will return $SS_{x_2} = \{\{x_4\}, \{x_5\}\}$), and P_1 will compute $SS_{x_1} = \{\{x_2, x_4\}, \{x_2, x_5\}\}$, and will return $Ans_{x_1} = Yes$, as

$\{x_2, x_4\}$ is the strongest mapping set in SS_{x_1} and P_4 precedes P_3 in T_1.

7.2 Properties of P2P_DR$_4$

Theorems 1 and 2 (regarding termination and total number of messages) hold also for this version, $P2P\_DR_4$ (the proof is exactly the same).

The main drawback of this approach is its too high computational complexity, which is the result of retaining all the different ways (supportive sets) that can be applied to support a literal. Specifically, the supportive set of a literal (SS_{x_i}) is the unification of the supportive sets of the rules that support it ($\bigcup SS_{r_i}$). The supportive set of each rule (SS_{r_i}) derives from the Cartesian Product of the Supportive Sets of the literals in its body. This means that if l is the number of literals contained in the body of a rule, and k_1 is the number of mapping sets in each literal's supportive set, SS_{r_i} will contain l_1^k different mapping sets. Considering that l is proportional to the number of literals a peer may define (n_l) and the number of acquaintances a peer may have (n_{ACQ}), the supportive set of a literal will contain $O(n_r^{(n_l \times n_{ACQ})^{k_1}})$ mapping sets. In the same way, k_1 will be $O(n_r^{(n_l \times n_{ACQ})^{k_2}})$ and so on. Overall, the number of distinct mapping sets that may be contained in the Supportive Set of a literal may be (in the worst case) exponential to the number of peers (n), to the number of rules a peer may define (n_r) and to the number of literals a peer may define (n_l), rendering this approach non-scalable and inapplicable even for a small number of peers.