Consistency Reasoning in Neural-Logic Belief Networks

Technical Report Number 467

July, 1993

Boon Toh Low and Norman Y Foo

ISBN 0 86758 684 2
Consistency Reasoning in Neural-Logic Belief Networks

Boon Toh LOW  
Norman Y. FOO
Knowledge Systems Group
Basser Department of Computer Science
University of Sydney
NSW, Australia 2006
boontoh@cs.su.oz.au  norman@cs.su.oz.au

Abstract

In this technical report, we argue that human belief can be contradictory and to maintain consistency for knowledge reasoning purposes, it is more natural to sort out conflicting beliefs by their relative strengths similar to how they are handled by our commonsense. Based on this philosophy, two notions of logical consistency (general and strong) that are semantically weaker than classical logical consistency are defined. We propose a method of reasoning and resolving these two forms of inconsistency for logical expressions in a three-valued belief representation system called Neural-Logic Belief Network. It is based on logical suppression where weaker beliefs are superseded by their stronger rivals and all suppressed beliefs are retained in the belief knowledge so that no information lost is due to this consistency reasoning process.

Keywords: consistency reasoning, three-valued knowledge representation, beliefs, philosophical foundations.
1 How Do We Deal With Contradictory Beliefs?

In symbolic knowledge representation systems, consistency of knowledge is one of the most important conditions to be upheld because no proper reasoning can be carried out if a knowledge/belief state is contradictory. For example, in Closed World Assumption systems (e.g. [Min82]) and AGM Logic (e.g. [AGM85, Gar88]), consistency of the closures of theories is one of the fundamental criteria, and for Default Reasoning (e.g. [Rei80, Poo88]) and Circumscription Logics (e.g. [McC80, Lif87]), extensions of theories have also to be logically consistent. In these classical-logic based formalisms, in order to avoid triviality, it is necessary to maintain consistency by disallowing or expelling contradictory knowledge from the final state (set) of beliefs. However, the insistence on classical logical consistency in beliefs is psychologically questionable, and attempts (e.g. [Sla91]) to weaken it using non-classical logic have been proposed.

Generally, there are three ways where beliefs are obtained: input beliefs that we gather from some information sources directly, beliefs derived from other beliefs and beliefs inherited genetically. Information obtained from different sources is one likely reason for having contradictory beliefs. Derived beliefs could also be inconsistent if the rules or other beliefs for deriving at these beliefs are themselves contradictory (such as conflicting properties from multiple inheritance). Biologically inherited beliefs are generally consistent but they can be contradictory as natural evolution is not always perfect.

How do we cope with incomplete and inconsistent beliefs in our commonsense reasoning? It is plausible that contradictory beliefs are not as drastically resolved as in classical logic (e.g. no inconsistent beliefs are allowed in the knowledge systems, or, all contradictory beliefs are removed from the belief states). These beliefs usually have different reliability levels, or, belief degrees. For example, direct personal experience will surely carry a stronger belief degree compared with information given by others, especially persons who have not established credibility. Based on the strength of beliefs (whatever its exact psychological basis may be), if two beliefs contradict we generally would agree that the stronger belief will over-ride the weaker one. If there is no clear winner, then we have to accept the contradictory state for that particular belief and stop using it for further reasoning.

Suppose one morning you left a piece of chocolate cake in the refrigerator and when you returned home, it was gone. If you were staying with your two kids: Paul and Emily then in your mind you may have the belief “Paul ate the cake OR Emily ate the cake” (or “both of them shared the cake” as included in the OR relation). After gathering further information, if you believe that “Paul did not eat the cake” because he went to school early and came back later than you, and “Emily did not eat the cake” because she was sick and had no appetite, then you are in a contradictory belief state. To straighten out these inconsistent beliefs, you have to perform some kinds of consistency reasoning. If there is a possibility that someone else might consume the cake such that the belief “Paul ate the cake OR Emily ate the cake” is weaker than the other two beliefs, then you might temporarily suppress your suspect about them eating the cake. If, on the other hand, you suspect that Emily may not really be sick and the belief
for “Emily did not eat the cake” is now weaker than the other two beliefs, you may now conclude from the two stronger beliefs that “Emily ate the cake” if you want to have someone responsible for the missing cake (This is the strong notion of consistency reasoning). If you are prepared to give Emily the benefit of the doubt, then you may neither believe that “Emily ate the cake” nor “Emily did not eat the cake” (The general notion of consistency reasoning). If all the three beliefs are of equal certainty, then you will not be sure which are the more reliable facts and may have to temporarily not believe in any of them since they contradict each other. Note that human reasoning does not discard any of the input information and the resulting beliefs are derived from suppressions of weaker beliefs in the consistency resolution process.

It is along this line of commonsense reasoning that the Consistency Reasoning Process is developed for a propositional three-valued knowledge representation system called Neural-Logic Belief Network [LF92, LF93]. In contrast with classical logics, its network computation nature treats logical relations as co-operative and competitive inputs from the sub-expressions which is shown to be a desirable property for human-like consistency reasoning in later sections.

2 Neural-Logic Belief Networks

This section provides an outline of a belief representation called Neural-Logic Belief Network [LF93]. A Neural-Logic Belief Network is an acyclical directed graph with a neural network computation model. There are two types of nodes: input nodes and base nodes. Input nodes receive input beliefs and propagate them via directed input links to the relevant base nodes. Each node represents a proposition (a variable-free concept expressed in words) and has a node value to indicate its current belief state. To model incomplete and inconsistent input knowledge, input nodes are allowed to represent different views of the same proposition from different sources. But for base nodes, each proposition is uniquely represented by one of them and the collection of all base nodes represents the belief state, usually denoted as $S$.

Each node value is an ordered pair: $(\text{proposition-value}, \text{degree-of-belief value})$. There are three possibilities for the proposition-value $(t, f)$:

- $(1, 0)$ means that the proposition associated with the node is believed,
- $(0, 1)$ that the negation of the proposition is believed, and
- $(0, 0)$ that it is neither believed nor not-believed.

The degree-of-belief values, written as deg[$\text{The proposition of the node}$], represent the strength and/or certainty of the propositions concerned. This induces a total asymmetric order (TAO) on all the degree-of-belief values. For example, we may have a set of beliefs with the following order:

$0 < \text{possibly} < \text{likely} < \text{weakly} < \text{usually} < \text{strongly believed} < \text{definitely}$

If we let “0” denotes the weakest degree value, i.e. no belief at all, then $(0, 0)$ is the default node value of beliefs not represented by the network or unknown beliefs in the belief state. If the unknown belief is $a$ and the current belief state is $S$, then $a \notin S$ and $\neg a \notin S$. A base node $a$ having a node value of $((0, 0), \text{deg}[a])$ where $\text{deg}[a] > 0$ means
that this belief currently has contradictory inputs of the same strength as indicated by its degree-of-belief value \( \text{deg}[a] \). That is, both \( a \in S \) and \( \neg a \in S \); or we use \( \text{!a} \in S \) to denote the contradictory state of belief. A base node \( a \) having a node value of \( (1, 0) \), \( \text{deg}[a] \) means the proposition \( a \) is believed in the belief state \( S \), i.e. \( a \in S \); if its node value is \( ((0, 1), \text{deg}[a]) \) then \( \neg a \in S \). Node values of \( ((1, 0), 0) \) and \( ((0, 1), 0) \) do not have proper semantic interpretation and they are invalid.

Each directed link is associated with an ordered pair of link weights \((u, v)\), where \( u, v \in \{\text{Real Numbers}\} \). Analogous to proposition-values, the first weight \( u \) is an excitatory value and the second weight \( v \) is an inhibitory value. There are four types of links: \textit{combinative links}, \textit{inheritance links}, \textit{input links} and \textit{rule links}.

Logical expressions are constructed by \textit{combinative links} and they follow Kleene's strong three-valued logic [Kle64]. For example, a two input OR relation \( a \lor b \) and a two input AND relation \( c \land d \) are represented as:

![Diagram of OR and AND relations](attachment:diagram.png)

where \( (2, 1/2) \) and \( (1/2, 2) \) are the corresponding link weights and the truth tables are given below:

<table>
<thead>
<tr>
<th>( a )</th>
<th>( b )</th>
<th>( a \lor b )</th>
<th>( \text{deg}[a \lor b] )</th>
<th>( a \land b )</th>
<th>( \text{deg}[a \land b] )</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1, 0)</td>
<td>(1, 0)</td>
<td>(1, 0)</td>
<td>( \max(\text{deg}[a], \text{deg}[b]) )</td>
<td>(1, 0)</td>
<td>( \min(\text{deg}[a], \text{deg}[b]) )</td>
</tr>
<tr>
<td>(1, 0)</td>
<td>(0, 0)</td>
<td>(1, 0)</td>
<td>( \text{deg}[a] )</td>
<td>(0, 0)</td>
<td>( \text{deg}[b] )</td>
</tr>
<tr>
<td>(1, 0)</td>
<td>(0, 1)</td>
<td>(1, 0)</td>
<td>( \text{deg}[a] )</td>
<td>(0, 1)</td>
<td>( \text{deg}[b] )</td>
</tr>
<tr>
<td>(0, 0)</td>
<td>(1, 0)</td>
<td>(1, 0)</td>
<td>( \text{deg}[b] )</td>
<td>(0, 0)</td>
<td>( \text{deg}[a] )</td>
</tr>
<tr>
<td>(0, 0)</td>
<td>(0, 0)</td>
<td>(0, 0)</td>
<td>( \max(\text{deg}[a], \text{deg}[b]) )</td>
<td>(0, 0)</td>
<td>( \max(\text{deg}[a], \text{deg}[b]) )</td>
</tr>
<tr>
<td>(0, 0)</td>
<td>(0, 1)</td>
<td>(0, 0)</td>
<td>( \text{deg}[a] )</td>
<td>(0, 1)</td>
<td>( \text{deg}[b] )</td>
</tr>
<tr>
<td>(0, 1)</td>
<td>(1, 0)</td>
<td>(1, 0)</td>
<td>( \text{deg}[b] )</td>
<td>(0, 1)</td>
<td>( \text{deg}[a] )</td>
</tr>
<tr>
<td>(0, 1)</td>
<td>(0, 0)</td>
<td>(0, 0)</td>
<td>( \text{deg}[b] )</td>
<td>(0, 1)</td>
<td>( \text{deg}[a] )</td>
</tr>
<tr>
<td>(0, 1)</td>
<td>(0, 1)</td>
<td>(0, 1)</td>
<td>( \min(\text{deg}[a], \text{deg}[b]) )</td>
<td>(0, 1)</td>
<td>( \max(\text{deg}[a], \text{deg}[b]) )</td>
</tr>
</tbody>
</table>

where the functions \( \max(\ldots) \) and \( \min(\ldots) \) return the maximum value and minimum value of the arguments respectively. For an \( n \)-input OR, each link weight is \( (2, \frac{1}{n}) \) and for \( n \)-input AND relations, the link weights are \( \left(\frac{1}{n}, 2\right) \) [LF93].

![Diagram of OR and AND relations](attachment:diagram.png)

Typically, the \textit{combined input proposition value} \((t_o, f_o)\) to a logical expression base node \( o \) (as shown in the above figure) with inputs from \( n \) sub-expressions \((o_1, o_2, \ldots, o_n)\), where \( 0 < i < n \) via \( n \) combinative links, where each sub-expression has a proposition-value of \((t_i, f_i)\) and a link weight of \((u_i, v_i)\), is given by the following \textit{Thresholding}
Function:

\[
(t_o, f_o) = \begin{cases} 
(1, 0) & \text{if } NET \geq 1 \\
(0, 1) & \text{if } NET \leq -1 \\
(0, 0) & \text{otherwise}
\end{cases}
\]

where the value of \( NET \) is given by the following Transfer Function:

\[
NET = \left( \sum_{i=1}^{n} E_{o_i} \right) - \left( \sum_{i=1}^{n} I_{o_i} \right)
\]

where \( E_{o_i} \in E \), which is the set of excitatory inputs (i.e. inputs for the proposition \( o \)):

\[
E = \left\{ E_{o_i} \mid E_{o_i} = |f_i \times v_i| \text{ where } f_i \times v_i < 0, \text{ otherwise } \right. \\
\left. E_{o_i} = t_i \times u_i \text{ where } t_i \times u_i \geq 0, \text{ for all } 0 < i \leq n \right\}
\]

and \( I_{o_i} \in I \), which is the set of inhibitory inputs (i.e. inputs against the proposition \( o \)):

\[
I = \left\{ I_{o_i} \mid I_{o_i} = |f_i \times u_i| \text{ where } t_i \times u_i < 0, \text{ otherwise } \right. \\
\left. I_{o_i} = f_i \times v_i \text{ where } f_i \times v_i \geq 0, \text{ for all } 0 < i \leq n \right\}
\]

The input degree-of-belief value for node \( o \), \( \text{deg}[o] \), depends on the computed combined input proposition value.

- If the computed combined input proposition value of node \( o \) using the above computation functions is \((0, 0)\), which means that neither the excitatory inputs nor the inhibitory inputs are strong enough to suppress each other. In these cases, the degree-of-belief value of node \( o \) is

\[
\text{deg}[o] = \begin{cases} 
\max(\text{deg}[o_i]) & \text{for any } t_i = 0 \text{ and } f_i = 0 \text{ where } 0 < i \leq n, \\
0 & \text{otherwise.}
\end{cases}
\]

where the function \( \max(\ldots) \) returns the maximum value among the arguments according to the total order of degree-of-belief values.

- If the computed combined input proposition value of node \( o \) is \((1, 0)\), we know that the contribution from excitatory inputs \((E_{o_i} \in E)\) to the node is stronger than that from the inhibitory inputs \((I_{o_i} \in I)\). The degree-of-belief of node \( o \), \( \text{deg}[o] \), is thus contributed from the corresponding \( \text{deg}[o_i] \) of the member \( E_{o_i} \) from \( E \) as follow:

\[
\text{deg}[o] = \begin{cases} 
\text{deg}[o_k] & \text{if } E_{o_k} > 1, \text{deg}[o_k] \geq \text{deg}[o_i] \text{ for all } 0 < i \leq n, \\
E_{o_k}, E_{o_i} \in E \text{ and } i \neq k, \text{ otherwise} \\
\text{deg}[o_i] & \text{if } \text{Min}[(\sum_{j=1}^{k} E_{o_j} \geq 1), \text{deg}[o_j] \geq \text{deg}[o_{j+1}]], \\
& \text{deg}[o_j] \geq \text{deg}[o_i], E_{o_j}, E_{o_i} \in E, \\
& \text{for all } 0 < i \leq n \text{ where } i \neq j.
\end{cases}
\]

where \( \text{Min}[(\sum_{j=1}^{k} E_{o_j} \geq 1)] \) is the minimum value of the sum \( \sum_{j=1}^{k} E_{o_j} \) that is greater or equal to 1 summed by the descending order of the corresponding \( \text{deg}[o_j] \), in which \( \text{deg}[o_j] \geq \text{deg}[o_{j+1}] \).
Figure 1: The degree-of-belief values of conjunction and disjunction.

- If the computed combined input proposition value of node \( o \) is \((0, 1)\), this indicates that the contribution from inhibitory inputs is greater than the inputs from excitatory inputs. The degree-of-belief of node \( o \) is computed by the last formula by replacing all the “E” to “T” accordingly.

As each logical expression is uniquely represented by a base node, only logical expressions in *Conjunctive Normal Form*\(^1\) (CNF) are accepted in the belief network to represent their equivalence.

**Proposition 1 (Degree-of-belief values of disjunctions and conjunctions)** Using the network computation functions defined, the two basic logical connectives for constructing logical expressions, \( \text{AND}(\wedge) \) and \( \text{OR}(\vee) \), will have the minimum and maximum degree-of-belief values from their sub-expressions respectively, if all sub-expressions are believed. \( \square \)

In figure 1, the horizontal axis represents increasing degree-of-belief values and the markings on the axis indicate where the associated propositions are located in the total asymmetric order (TAO). Here, we have the ranking of the degree-of-belief values in ascending order for three beliefs as \( \text{deg}[a] < \text{deg}[b] < \text{deg}[c] \); we also have \( \text{deg}[a \wedge b \wedge c] = \text{deg}[a] \) and \( \text{deg}[a \vee b \vee c] = \text{deg}[c] \).

Input-links, inheritance-links and rule-links are “alternative” links. The values propagated by these links are not combined with other links. They are instead considered as possible alternative input values to the node concerned and are each computed using the same computation functions for combinative-links above. When there are \( n \) inputs from \( n \) alternative links to a base node \( o \), with the network input values from each input \(((t_i, f_i), \text{deg}[i]) \ (0 < i < \leq n)\), the following *Selection Function* is used to select the strongest inputs according to the TAO as the node value for \( o \).

\[
(t_o, f_o) = \begin{cases} (t_k, f_k) & \text{if } \text{deg}[k] > \text{deg}[i], 0 < k \leq n, k \neq i, \text{ for all } 0 < i \leq n \\ (t_j, f_j) & \text{if } (t_j, f_j) = (t_k, f_k), 0 < j \leq m, 0 < k \leq m, 1 < m \leq n, \\ & \text{deg}[j] = \text{deg}[k], \text{deg}[j] > \text{deg}[i], j \neq k \neq i, \text{ for all } 0 < i \leq n \\ (0, 0) & \text{otherwise} \end{cases}
\]

\[
\text{deg}[o] = \begin{cases} \text{deg}[k] & \text{if } \text{deg}[k] \geq \text{deg}[i], 0 < k \leq n, k \neq i, \text{ for all } 0 < i \leq n \\ 0 & \text{otherwise} \end{cases}
\]

\(^1\)For example, \( A_1 \wedge A_2 \wedge \ldots \wedge A_n, n > 0 \) and each \( A_i \ (0 < i < n) \) is a either a literal (proposition) or a pure disjunctive logical expression. Having this conjunction is at least equivalent to having all conjuncts, i.e. \( A_1, A_2, \ldots, A_n \), at the same degree-of-belief value as \( \text{deg}[A_1 \wedge A_2 \wedge \ldots \wedge A_n] \) in the belief state [LF92].
The node value of a logical expression is usually computed from its sub-expressions using the transfer and thresholding functions above. However, direct inputs to logical expressions via input links, inheritance links and rule links are possible in belief networks. In these cases, the input values from all combinative-links shall first be processed to yield a combined input value (i.e. a combined input proposition value and input degree-of-belief value). This node value is treated as one of the alternative input values and the node value of the logical expression is computed using the selection function defined.

This selection mechanism provides the belief network with a tolerance for inconsistencies, such as may arise between an expression and its sub-expressions. Take for example, beliefs \( a \ ((0, 1), \text{weakly}) \), \( b \ ((0, 1), \text{weakly}) \), a direct input \( a \lor b \ ((1, 0), \text{strongly}) \) and another input from a rule link to \( a \lor b \ ((0, 1), \text{usually}) \) are in the belief network, the node value of this logical expression \( a \lor b \) is selected among its conflicting alternative inputs; and since the direct input has the strongest degree-of-belief value, the selected node value for \( a \lor b \) is \((1, 0), \text{strongly}\). These computation functions resolve inconsistencies at each proposition (at each base node representing the proposition) locally. They lay the foundation for modeling human-like conflict resolution in consistency reasoning.

Other than logical relations, non-classical logical relations (relations that may give rise to nonmonotonic characteristics) such as the defeasible IF-THEN rules represented by rule links, inheritance relations modeled using inheritance links, etc., are discussed in [LF93, Low93].

There is a set of six Network Update Operators for the belief network [LF92]. The operators Add and Update are used to include new beliefs into or update existing beliefs of a belief state. Remove and Forget are used to expel existing beliefs, and Not-Conclude is used to temporarily suppress a belief to a proposition-value of \((0, 0)\). The other operator Revise is for making revision to the belief state according to a new belief and it is based on the Add and Remove operators. When there is inconsistency after a network update operation, consistency reasoning is necessary.

3 Resolving Inconsistent Beliefs

In Neural-Logic Belief Networks, we adopt the following three principles for consistency reasoning:

1. Minimal change to existing beliefs — This will ensure that the least possible number of beliefs are affected when resolving inconsistencies.
2. Changes start from the weakest belief — When there is inconsistency in the belief state, changes to beliefs made by the consistency reasoning process always start from the weakest belief(s) concerned. The relative strengths of beliefs are determined by the TAO of degree-of-belief values associated with the beliefs.
3. Suppression of beliefs — the weakest inconsistent beliefs affected in consistency reasoning are suppressed by their stronger counterparts and they remain in the belief knowledge rather than being expelled from the beliefs.
These principles correspond to what we believe as the essential properties of human consistency resolution. They are different from classical logical consistency where only relations following strictly from the truth tables are permitted in any logical belief set.

Throughout this section, literals or the name of simple propositions without any logical connectives attached, are represented by italic lower-case English alphabets such as $a$, $b$, $c$, $d$, ..., etc. Upper-case Greek letters such as $\Psi$, $\Theta$, ..., etc. are used to denote CNF logical expressions while their sub-expressions with proposition-values, for example $a$ $(1, 0)$, $b$ $(0, 1)$ (which means $\neg b$), $c \lor d$ $(1, 0)$, etc., are represented by lower-case Greek letters such as $\alpha$, $\beta$, $\gamma$, $\delta$, ..., etc. Sub-scripts and/or super-scripts may be attached to these notations.

The next sub-section defines the two notions of consistency: General and Strong. Under each of these definitions and the above three principles, the main Consistency Reasoning Process is formalized using the three sub-processes: Relaxation of Logical Suppression (sub-section 3.3), Consistency Check (sub-section 3.4) and Consistency Maintenance (sub-section 3.5). Sub-section 3.2 discusses the main building block for consistency maintenance: Logical Suppression. Sub-section 3.6 summarizes a procedure for carrying out the human-like consistency reasoning in a belief network and provides examples to illustrate differences between the two notions of consistency reasoning. The last sub-section (3.7) discusses the mixing of G- and S- consistency for different beliefs.

### 3.1 Two Notions of Consistencies: General and Strong

In this section, we formally define the two notions of human-like consistency reasoning described in the introductory section. First, we need some basic definitions.

Given a logical expression base node $\Psi$ with $n$ sub-expressions, any of its sub-expressions $\alpha_i$ $(0 < i \leq n)$ with a proposition-value $(t_i, f_i)$ and a link weight of $(u_i, v_i)$ has an input from $\alpha_i$ to $\Psi$, $(x, y)$, where $(x, y) = (t_i \times u_i, f_i \times v_i)$. This input is a contributing agreeing input where

- $x > 0$ or $y < 0$ if the proposition-value of $\Psi$ is $(1, 0)$; or,
- $x < 0$ or $y > 0$ if the proposition-value of $\Psi$ is $(0, 1)$.

This input is a contributing disagreeing input where

- $x < 0$ or $y > 0$ if the proposition-value of $\Psi$ is $(1, 0)$; or,
- $x > 0$ or $y < 0$ if the proposition-value of $\Psi$ is $(0, 1)$.

A sub-expression agrees with a logical expression if and only if it is a contributing agreeing input to the proposition-value of the logical expression. It disagrees with a logical expression if and only if it is a contributing disagreeing input to the proposition-value of the logical expression.

For example, a proposition $a$ with a node value of $((0, 1), \text{deg}[a])$ agrees with the logical expression $a \lor b$ $((0, 1), \text{deg}[a \lor b])$ and it disagrees with the belief $a \lor b$ $((1, 0), \text{deg}[a \lor b])$.

Following from the computation functions for logical expressions and proposition 1, for
A conjunctive belief where at least one conjunct has a higher degree-of-belief value than the conjunction.

By the logical relation defined, \( \neg (a \land \neg b \land \neg c \land \neg d) \) is in the belief state.

That is, \( \neg (a \lor b \lor c \lor d) \) is believed at the weakest degree-of-belief value of the opposing disjunct.

A disjunctive belief where all disagreeing disjuncts have higher degree-of-belief value than the disjunction.

By the logical relation defined, \( (\neg a \lor \neg b \lor \neg c \lor \neg d) \) is in the belief state.

The conjunctive relation will be suppressed as long as at least one of its opposing conjunct(s) is stronger than it.

Figure 2: Conjunctive and disjunctive beliefs with stronger disagreeing sub-expressions.

A disjunction, if all its sub-expressions disagree with it and they all have higher degree-of-belief values than the disjunction, its negation will be believed. For a conjunction, as long as there is a stronger disagreeing sub-expression, the negation of the conjunction will be believed. Figure 2 illustrates these two situations where boxed beliefs are suppressed by stronger beliefs in the belief state.

A set of sub-expressions is said to be non-interfering with its logical expression if and only if both the combined input proposition value of this set and the proposition-value of the logical expression are \((0, 0)\):

<table>
<thead>
<tr>
<th>The proposition-value of the logical expression</th>
<th>The combined input proposition value computed from a set of sub-expressions</th>
</tr>
</thead>
<tbody>
<tr>
<td>((0, 0))</td>
<td>((0, 0))</td>
</tr>
</tbody>
</table>

A logical expression is said to be supported by a set of its sub-expressions if and only if the computed combined input proposition value from these sub-expressions is not \((0, 0)\) and it has the same proposition-value as the logical expression as shown below:

<table>
<thead>
<tr>
<th>The proposition-value of the logical expression</th>
<th>The combined input proposition value computed from a set of sub-expressions</th>
</tr>
</thead>
<tbody>
<tr>
<td>((1, 0))</td>
<td>((1, 0))</td>
</tr>
<tr>
<td>((0, 1))</td>
<td>((0, 1))</td>
</tr>
</tbody>
</table>

Otherwise, it is a set of non-supporting sub-expressions. For example, \(a \lor b ((1, 0), \text{deg}[a \lor b])\) is supported by \(a ((1, 0), \text{deg}[a])\), or \(b ((1, 0), \text{deg}[b])\). \(a \land b ((1, 0), \text{deg}[a \land b])\) is supported by \(a ((1, 0), \text{deg}[a])\) and \(b ((1, 0), \text{deg}[b])\). Figure 3 shows some examples where logical expressions are supported by their sub-expressions.

A logical expression is opposed by a set of its sub-expressions if and only if the combined input proposition value of the set of sub-expressions is neither \((0, 0)\) nor supporting the
logical expression:

<table>
<thead>
<tr>
<th>The proposition-value of the logical expression</th>
<th>The combined input proposition value computed from a set of sub-expressions</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1, 0)</td>
<td>(0, 1)</td>
</tr>
<tr>
<td>(0, 1)</td>
<td>(1, 0)</td>
</tr>
<tr>
<td>(0, 0)</td>
<td>(1, 0)</td>
</tr>
<tr>
<td>(0, 0)</td>
<td>(0, 1)</td>
</tr>
</tbody>
</table>

For example, the belief $a \lor b$ ((1, 0), deg$[a \lor b]$) is opposed by $a$ ((0, 1), deg$[a]$) and $b$ ((0, 1), deg$[b]$).

Illustrations of the non-interfering set, support set and oppose set are shown in Figure 4.

**Definition G-Consistency** (The General Notion of Consistency)
Given a logical expression, when the combined input value from all its sub-expressions opposes it, it is **G-Inconsistent** (with its sub-expressions); otherwise, it is **G-Consistent**.

Under this definition and the three principles, any inconsistency is resolved by suppressing the group of weakest opposing sub-expressions to a proposition-value of (0, 0) so that their opposition to the logical expression becomes insignificant and the beliefs are G-consistent.

**Definition S-Consistency** (The Strong Notion of Consistency)
In addition to the general notion of consistency, when there exist a unique weakest non-supporting and not non-interfering (figure 4) sub-expression $\beta$ from a logical expression $\Psi$ where deg$[\beta] <$ deg$[\Psi]$, and, all other sub-expressions of $\Psi$ disagree with $\Psi$, it is...
This type of consistency reasoning demands that if there exists a non-supporting weakest inconsistent sub-expression, make it S-consistent by suppressing it to support the logical expression. It requires a more forceful change than G-consistency reasoning. G-consistency is a general case of S-Consistency, if it is S-consistent then it is G-consistent. Note that both G- and S-consistency are more general than three-valued logical consistency which only includes the non-interfering set and support set as shown in figure 4.

For example, if we believe in \( a_1 \lor a_2 \lor \ldots \lor a_n \), where \( n > 1 \), the negation of \( a_2 \) to \( a_n \) (i.e. \( \neg(a_2 \lor \ldots \lor a_n) \)) and \( a_1 \) is the unique weakest disagreeing belief when compared to other sub-expressions that does not support \( a_1 \lor a_2 \lor \ldots \lor a_n \) (either with a proposition-value of \((0, 0)\), or \((0, 1)\)), then under the general notion of consistency,

- it is consistent if \( a_1 \) has a proposition-value of \((0, 0)\); or,
- if \( a_1 \) has a proposition-value of \((0, 1)\) then it has to be suppressed to \((0, 0)\) so that the belief state is consistent.

But under the strong notion, to maintain consistency, \( a_1 \) has to be suppressed to \((1, 0)\). This is the same as the *modus ponens* inference rule in classical logic given that the conclusion (e.g. \( a_1 \)) is a unique weakest literal.

\[
\begin{align*}
a_1 \lor a_2 \lor \ldots \lor a_n \\
\neg(a_2 \lor \ldots \lor a_n)
\end{align*}
\]

\[
\frac{a_1 \lor a_2 \lor \ldots \lor a_n \lor \neg(a_2 \lor \ldots \lor a_n)}{a_1}
\]

### 3.2 Logical Suppression

Suppressions come naturally with the computation functions of Neural-Logic Belief Networks. Any proposition associated with a node in a belief network could receive inputs from various sources and they can be mutually contradictory. Due to the competitive and co-operative nature of the computation model as shown in section 2, only the strongest or un-opposed inputs are selected at each base node locally into the current belief state. Although the weaker disagreeing inputs remain in the belief knowledge, but they are temporarily being suppressed by the stronger ones. In consistency reasoning, a similar minimal change principle is adopted where weaker inconsistent beliefs are suppressed rather than expelled from the belief knowledge.

To suppress a belief \( a \) with any initial proposition-value to a proposition-value of \((1, 0)\) (or \((0, 1)\)), a suppression input of \((1, 0)\) (correspondingly \((0, 1)\)) with a degree-of-belief value stronger than the existing \( \text{deg}[a] \) is introduced. To suppress the belief \( b \) to a proposition-value of \((0, 0)\), we introduce an opposite input, i.e. \((1, 0)\) for \((0, 1)\) or \((0, 1)\) for \((1, 0)\), with the same degree-of-belief value \( \text{deg}[b] \). If the initial proposition-value of \( b \) is \((0, 0)\), then it has already satisfied the intended suppression and no suppression is required.

Since consistency of inputs (direct inputs and input from the sub-expressions) to a logical expression is handled by the computation functions of belief networks, the only
possibility where inconsistency can occur is when the combined input value from all
the sub-expressions is weaker than and inconsistent with the logical expression, and
the current node value of the logical expression is derived from other stronger direct
input(s). Thus, to maintain consistency, suppressions are required from the logical
expressions to their sub-expressions, and they are called \textit{Logical Suppressions}. A logical
suppression from a logical expression making a sub-expression to a proposition-value
of either \((1, 0)\) or \((0, 1)\) will turn it into an supporting input to the logical expression.
This is called the \textit{Supporting Logical Suppression}. A logical suppression to a value of
\((0, 0)\) does not make the sub-expression into a supporting input and it is called \textit{Non-
Supporting Logical Suppression}. From a logical expression, only one suppression per
sub-expression is allowed and it is either a supporting or non-supporting one.

Given a logical expression \(\Psi\) and if its sub-expression \(\alpha\) is logically suppressed to \(\beta\) with
a degree-of-belief value of \(\text{deg}[\beta]\), where \(\alpha\) and \(\beta\) are the same proposition with different
proposition values (e.g. \(\alpha\) and \(\neg\alpha\)), it is denoted as:

\[
\alpha \xrightarrow{\Psi} (\beta, \text{deg}[\beta])
\]

Theoretically, for \(\beta\) to support its logical expression \(\Psi\), \(\text{deg}[\beta]\) can be anywhere along
the TAO except “0”. To be unbiased and conservative, \(\text{deg}[\Psi]\) is used as the reference
(maximum) point and all supporting logical suppressions are assigned to this value:

\[
\alpha \xrightarrow{\Psi} (\beta, \text{deg}[\Psi])
\]

But for any non-supporting logical suppression, \(\text{deg}[\beta]\) is assigned to the degree value
of the respective suppressed sub-expressions \(\text{deg}[\alpha]\):

\[
\alpha \xrightarrow{\Psi} (\beta, \text{deg}[\alpha])
\]

Since the strength of any logical suppression is limited by its logical expression \(\Psi\), we
have \(\text{deg}[\alpha] \leq \text{deg}[\Psi]\). Following from the above definitions that only one suppression
per sub-expression is allowed for each logical expression, we have the following proposition:

\textbf{Proposition 2} The maximum number of logical suppressions that can be imposed
from a logical expression is equal to the number its sub-expressions. \(\square\)

When all possible logical suppressions have been imposed and no more suppression is
possible because some of the opposing sub-expression have higher degree-of-belief values
than the logical expression, it is say to be \textit{exhaustively suppressed}. A special case is
when all the sub-expressions of a logical expression are logically suppressed, it is said to
have \textit{full suppression} or it is \textit{fully suppressed}. A fully suppressed belief is both G- and S-
consistent (see Appendix). A logical suppression from a logical expression \(\Psi\), \(\alpha \xrightarrow{\Psi} (\beta,
\text{deg}[\beta])\), is \textit{counter-suppressed} when it has been suppressed by another stronger logical
suppression to a belief inconsistent with \(\Psi\). That is there exist at least another logical
suppression \(\alpha \xrightarrow{\Theta} (\gamma, \text{deg}[\gamma])\) where \(\gamma \neq \beta\), \(\text{deg}[\gamma] > \text{deg}[\beta]\) and \(\gamma\) is inconsistent with
\(\Psi\). If a belief is removed from the belief knowledge, all its logical suppressions are also
removed.
3.3 Relaxation of Logical Suppression

When the sub-expressions are having supporting logical suppressions, they can be supporting more than one logical expression simultaneously. For example, a supporting logical suppression of $\neg a \implies (a, \deg[\Psi])$ will make $a$ be a supporting input to $a \lor b$, $a \lor c \lor d$, etc. Sometimes one logical expression is supported by more than one supporting logical suppression (this situation may arise from many possibilities such as addition or removal of some beliefs), for example, $a \lor b$ can be supported by $\neg a \implies (a, \deg[\Psi])$ as well as $\neg b \implies (b, \deg[\Theta])$ at the same time although one of them is sufficient to maintain consistency. Some of these logical suppressions are considered as redundant and they have to be prevented to up-keep the minimal change principle.

A redundant logical suppression is one that can be removed from the belief state without changing the proposition-values of the logical expressions originally supported by it. In the $a \lor b$ example, either $\neg a \implies (a, \deg[\Psi])$ or $\neg b \implies (b, \deg[\Theta])$ can be removed. The sub-expressions with the redundant suppressions are considered to be over-suppressed and to prevent this from happening, they have to be removed from the beliefs and the removal of these logical suppressions is called Relaxation. Formally, a logical suppression $\alpha \implies (\beta, \deg[\beta])$ is relaxed when it is removed from the belief state and allows the sub-expression to regain its original belief status of $\alpha$ if possible (e.g. if $\alpha$ is not counter-suppressed by other stronger beliefs) because all beliefs suppressed by the consistency reasoning process remain in the belief knowledge. The logical suppressions of any exhaustively suppressed beliefs will not be relaxed as there is no redundancy among these suppressions. They are either not supporting the belief, or, only the last logical suppression is currently supporting it.

Logical relaxations are carried out under the following situations (whenever there is a redundant logical expression during the consistency reasoning process) to remove redundant logical expressions:

1. From a sub-expression $\alpha$, if all the supporting suppressions to it are of the same proposition-value, that is, $\alpha \implies (\alpha, \deg[\Psi])$, $\alpha \implies (\alpha, \deg[\Theta])$, ..., for any $\Psi$, $\Theta$, and they are the only supporting suppressions from $\Psi$, $\Theta$, and, by removing these suppressions, all these logical expressions are still supported by $\alpha$, then relax these suppressions.

\begin{figure}[h]
\centering
\begin{tikzpicture}
\node (a) at (0,0) {$a$};
\node (b) at (1,-1) {$\neg b$};
\node (c) at (2,0) {$a \lor c$};
\node (d) at (3,0) {$a \lor b$};
\node (deg1) at (4,-1) {$\deg[\Psi]$};
\node (deg2) at (5,-1) {$\deg[\Theta]$};
\node (relaxed) at (6,0) {RELAXED};
\draw[->] (a) -- (b);
\draw[->] (b) -- (c);
\draw[->] (c) -- (d);
\draw[->] (d) -- (deg1);
\draw[->] (a) -- (deg2);
\end{tikzpicture}
\end{figure}

In the figure above, the example shows that logical expressions $a \lor c$ is supported by $a \implies (a, \deg[a \lor c])$ and $a \lor b$ is supported by $a \implies (a, \deg[a \lor b])$. Both of these logical suppressions can be relaxed and after relaxation, $a \lor c$ and $a \lor b$ are still supported by $a$.

2. From a logical expression $\Psi$, if it is supported by more than one supporting sup-
pression from itself, that is, $\alpha_1 \xrightarrow{\Psi} (\beta_1, \deg[\Psi])$, $\alpha_2 \xrightarrow{\Psi} (\beta_2, \deg[\Psi])$, ..., etc., except the supporting suppression to the unique weakest sub-expression, relax all other supporting suppressions.

For the example in the above figure, logical expressions $a \lor b$ is supported by both $\neg a \xrightarrow{a \lor b} (a, \deg[a \lor b])$ and $\neg b \xrightarrow{a \lor b} (b, \deg[a \lor b])$. Since $\neg a$ is the unique weakest sub-expression being suppressed, the other supporting logical suppression, $\neg b \xrightarrow{a \lor b} (b, \deg[a \lor b])$, is relaxed.

3. From a logical expression $\Psi$ with $n$ logical suppressions, if it is supported by at least one of its sub-expression that are not suppressed by itself, then relax all suppressions from $\Psi$, that is, relax all $\alpha_i \xrightarrow{\Psi} (\beta_i, \deg[\beta_i])$ where $0 < i \leq n$.

In the above figure, logical expressions $a \lor b \lor c$ is supported by $c$ and the suppression $\neg a \xrightarrow{a \lor b \lor c} (a, \deg[\neg a])$ is relaxed.

### 3.4 Consistency Check

A Consistency Check is a checking process to find out inconsistent logical expression(s) in the belief state depending on the notion of consistency used (either G-consistency or S-consistency). This process is initiated by any belief update operations where belief change may take place. For each logical expression affected by a belief change process and for each of its sub-expressions, consistency checking is carried out to all logical expressions that are related to each of the sub-expressions. This means all logical expressions formed in-part by the sub-expression or its negations are tracked via the combination-links from the sub-expression. For example, given a sub-expression $a$, its related logical relations are any logical relations containing $a$ or $\neg a$ (e.g. $a \lor b$, $(\neg a \lor c) \land d$, etc.).

During the consistency reasoning process, consistency checks are carried out to the related logical expressions for every sub-expression suppressed by the current round of consistency reasoning to find out which of them is inconsistent.

### 3.5 Consistency Maintenance

Consistency Maintenance is a set of procedures for maintaining consistency of belief
states. These procedures do not make any changes to the beliefs if the belief state is consistent. When a logical expression is found to be inconsistent, this set of procedures force changes to make the belief state consistent by introducing appropriate logical suppression and relaxations. The principles for consistency reasoning: minimal change and changes always start from the weakest related beliefs, underly all consistency maintenance procedures. The consequence of an application of consistency maintenance is that a set of existing beliefs (possibly empty) will be logically suppressed, this is called the Suppressed Set denoted by $CS$, and a new set of beliefs (possibly empty) is evolved, this is called the Evolved Set denoted by $CE$, to replace the suppressed beliefs at the new belief state.

$$CS = \{ (\alpha, \deg(\alpha)) \mid \text{for all } \alpha \xrightarrow{\psi} (\beta, \deg(\beta)) \text{ in the consistency maintenance process} \}$$

$$CE = \{ (\gamma, \deg(\gamma)) \mid \text{for all } \alpha \xrightarrow{\psi} (\beta, \deg(\beta)) \text{ in each round of the consistency maintenance process} \}$$

and $\gamma$ is the final proposition for $\alpha$.

Following the definitions of G-inconsistency and S-inconsistency, we also have two types of consistency maintenance: the general notion of consistency maintenance and the strong notion of consistency maintenance. We shall note the differences as we go through the consistency maintenance process. As Neural-Logic Belief Networks only accept logical expressions in conjunctive normal form, only logical expressions formed by the two basic logical connectives AND ($\land$) and OR ($\lor$) need to be examined.

To begin, let us look at a two-input disjunctive expression $a \lor b$ to illustrate how consistency maintenance is carried out to all the possible situations. Table 1 shows the exhaustive list of permutations for the proposition-values for $a$, $b$ and $a \lor b$ with the remarks column indicating the main consistency maintenance procedures if it is inconsistent. In this table, rows marked with a “+” sign indicate the values correspond to the truth table for a two-input OR relation (page 4), which are logically consistent (and therefore G- and S-consistent). Those marked with a “*” indicate that they do not correspond to the truth table but are G-consistent and those marked with a “**” means they are both G- and S-consistent. Those marked with a “@” indicate that they are G-inconsistent and those marked with a “=” indicate that they are S-inconsistent. Those without any marking but have a “—not applicable—” label in their remarks columns indicate that they are not the possible outcomes for belief networks.

Consistency maintenance for a pure disjunctive logical expression (not necessarily restricted to a two-input disjunction) are carried out under the following general categories corresponding to cases in table 1:

1. When a disjunctive logical expression is believed and the values of its sub-expressions correspond to that of a truth table, the belief states are logically consistent and therefore G- and S-consistent. (This corresponds to cases 1 to 4, and 7 of the $a \lor b$ example in table 1)

2. When a disjunction is believed and if all its sub-expressions have a proposition-value of $(0, 0)$, it is both G-consistent and S-consistent. (case 5)
<table>
<thead>
<tr>
<th>Case</th>
<th>(a)</th>
<th>(b)</th>
<th>(a \lor b)</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 +</td>
<td>(1, 0)</td>
<td>(1, 0)</td>
<td>(1, 0)</td>
<td>G- and S-consistent.</td>
</tr>
<tr>
<td>2 +</td>
<td>(1, 0)</td>
<td>(0, 0)</td>
<td>(1, 0)</td>
<td>G- and S-consistent.</td>
</tr>
<tr>
<td>3 +</td>
<td>(1, 0)</td>
<td>(0, 1)</td>
<td>(1, 0)</td>
<td>G- and S-consistent.</td>
</tr>
<tr>
<td>4 +</td>
<td>(0, 0)</td>
<td>(1, 0)</td>
<td>(1, 0)</td>
<td>G- and S-consistent.</td>
</tr>
<tr>
<td>5 **</td>
<td>(0, 0)</td>
<td>(0, 0)</td>
<td>(1, 0)</td>
<td>G- and S-consistent.</td>
</tr>
<tr>
<td>6 *</td>
<td>(0, 0)</td>
<td>(0, 1)</td>
<td>(1, 0)</td>
<td>G-consistent but could be S-inconsistent. — under S-consistency reasoning, suppress the weakest sub-expression to (1, 0).</td>
</tr>
<tr>
<td>7 +</td>
<td>(0, 1)</td>
<td>(1, 0)</td>
<td>(1, 0)</td>
<td>G- and S-consistent.</td>
</tr>
<tr>
<td>8 *</td>
<td>(0, 1)</td>
<td>(0, 0)</td>
<td>(1, 0)</td>
<td>G-consistent but could be S-inconsistent. — under S-consistency reasoning, suppress the weakest sub-expression to (1, 0).</td>
</tr>
<tr>
<td>9 @</td>
<td>(0, 1)</td>
<td>(0, 1)</td>
<td>(1, 0)</td>
<td>G-inconsistent and S-inconsistent. — under G-consistency reasoning, suppress the sub-expression(s) to (0, 0). — under S-consistency reasoning, suppress the weakest sub-expression to (1, 0).</td>
</tr>
<tr>
<td>10 @</td>
<td>(1, 0)</td>
<td>(1, 0)</td>
<td>(0, 0)</td>
<td>G- and S-inconsistent — suppress all sub-expressions to (0, 0).</td>
</tr>
<tr>
<td>11 @</td>
<td>(1, 0)</td>
<td>(0, 0)</td>
<td>(0, 0)</td>
<td>G- and S-inconsistent — suppress all sub-expressions to (0, 0).</td>
</tr>
<tr>
<td>12 @</td>
<td>(1, 0)</td>
<td>(0, 1)</td>
<td>(0, 0)</td>
<td>G- and S-inconsistent — suppress all sub-expressions to (0, 0).</td>
</tr>
<tr>
<td>13 @</td>
<td>(0, 0)</td>
<td>(1, 0)</td>
<td>(0, 0)</td>
<td>G- and S-inconsistent — suppress all sub-expressions to (0, 0).</td>
</tr>
<tr>
<td>14 +</td>
<td>(0, 0)</td>
<td>(0, 0)</td>
<td>(0, 0)</td>
<td>G- and S-consistent.</td>
</tr>
<tr>
<td>15 +</td>
<td>(0, 0)</td>
<td>(0, 1)</td>
<td>(0, 0)</td>
<td>G- and S-consistent.</td>
</tr>
<tr>
<td>16 @</td>
<td>(0, 1)</td>
<td>(1, 0)</td>
<td>(0, 0)</td>
<td>G- and S-inconsistent — suppress all sub-expressions to (0, 0).</td>
</tr>
<tr>
<td>17 +</td>
<td>(0, 1)</td>
<td>(0, 0)</td>
<td>(0, 0)</td>
<td>G- and S-consistent.</td>
</tr>
<tr>
<td>18 @</td>
<td>(0, 1)</td>
<td>(0, 1)</td>
<td>(0, 0)</td>
<td>G- and S-inconsistent — suppress all sub-expressions to (0, 0).</td>
</tr>
<tr>
<td>19</td>
<td>(1, 0)</td>
<td>(1, 0)</td>
<td>(0, 1)</td>
<td>—not applicable—</td>
</tr>
<tr>
<td>20</td>
<td>(1, 0)</td>
<td>(0, 0)</td>
<td>(0, 1)</td>
<td>—not applicable—</td>
</tr>
<tr>
<td>21</td>
<td>(1, 0)</td>
<td>(0, 1)</td>
<td>(0, 1)</td>
<td>—not applicable—</td>
</tr>
<tr>
<td>22</td>
<td>(0, 0)</td>
<td>(1, 0)</td>
<td>(0, 1)</td>
<td>—not applicable—</td>
</tr>
<tr>
<td>23</td>
<td>(0, 0)</td>
<td>(0, 0)</td>
<td>(0, 1)</td>
<td>—not applicable—</td>
</tr>
<tr>
<td>24</td>
<td>(0, 0)</td>
<td>(0, 1)</td>
<td>(0, 1)</td>
<td>—not applicable—</td>
</tr>
<tr>
<td>25</td>
<td>(0, 1)</td>
<td>(1, 0)</td>
<td>(0, 1)</td>
<td>—not applicable—</td>
</tr>
<tr>
<td>26</td>
<td>(0, 1)</td>
<td>(0, 0)</td>
<td>(0, 1)</td>
<td>—not applicable—</td>
</tr>
<tr>
<td>27 +</td>
<td>(0, 1)</td>
<td>(0, 1)</td>
<td>(0, 1)</td>
<td>G- and S-consistent.</td>
</tr>
</tbody>
</table>

Table 1: All possible cases for a two input OR relation \(a \lor b\).
3. When a disjunctive expression is believed, and, if its disjuncts consist of disjuncts with proposition-values of either (0, 1) or (0, 0) and at least one of them is (0, 0), it is G-consistent and there is no change to the belief state.

Under S-consistency maintenance, it is consistent except when there exist a unique weakest disjunct that is also weaker than the disjunction, then it is S-inconsistent. To maintain S-consistency, a supporting logical suppression is imposed to this disjunct to make it support the disjunctive logical expression. Consistency check and relaxation are carried out to the affected beliefs starting from the suppressed disjunct. (cases 6 and 8)

The following figure illustrates S-consistency maintenance for the $a \lor b$ example where the belief $a$ is the unique weakest inconsistent disjunct:

4. When a disjunction is believed and the negations of all it disjuncts are also believed, it is both G- and S-inconsistent. This could only happen when at least one of its disjuncts is weaker than the disjunction in the TAO.\(^2\)

To maintain G-consistency, the weakest sub-expression or the group of weakest sub-expressions is suppressed to (0, 0) at the same weakest degree-of-belief value. Consistency check and relaxation are carried out to the affected beliefs. (case 9)

The following figure shows G-consistency maintenance for $a \lor b$ with both $\neg a$ and $\neg b$ are believed at the same weaker degree value.

5. When neither the disjunction nor its negation is believed and the values of the sub-expression correspond to the definition of an OR relation (the truth table), the belief states are consistent. (cases 14, 15 and 17)

---

\(^2\)If all its disjuncts are stronger than the disjunction, the negation of the disjunction will be believed. If the weakest disjunct is of the same degree-of-belief value as the disjunction, the resulting disjunction will have a proposition-value of (0, 0).
6. When a disjunction has a proposition-value of \((0, 0)\) and the combine input proposition value from all its disjuncts opposes it, it is G- and S- inconsistent. To maintain consistency, each disjunct whose proposition-value is not \((0, 0)\) is suppressed with a non-supporting logical suppression so that the combined input proposition value of all disjuncts become \((0, 0)\). (cases 10 to 13, 16 and 18)

7. When the negation of a disjunction is believed, and the combine input proposition value all its disjuncts supports it, it is logically consistent (G- and S- consistent). (case 27)

8. When the negation of a disjunctive expression is introduced into the belief state, the negation of each of its disjuncts are also believed at the same degree-of-belief value as the disjunction (e.g. \(\neg(a \lor b \lor c)\) is equivalent to \(\neg a \land \neg b \land \neg c\)). From the computation functions, for the negation of the disjunction to be believed, the negation of all its disjuncts must also be believed. This makes all other combinations impossible and they need not be considered for consistency maintenance. (cases 19 to 26)

We have covered all possible consistency maintenance situations for a disjunctive expression. Since connective AND is logically symmetrical to OR in belief networks as shown by the truth table on page 4, consistency maintenance for conjunctive expressions are carried out in the same ways as the above categories by switching context from “disjuncts” to “conjuncts”, and, replacing \((1, 0)\) for \((0, 1)\) and \((0, 1)\) for \((1, 0)\) respectively (see table 2 below).

### 3.6 The Consistency Reasoning Process

The functions of the *Consistency Reasoning Process* are to reason about inconsistencies in belief states, to resolve inconsistencies by making minimum adjustments to beliefs (via logical suppression(s) to the weakest relevant belief(s)), and to prevent unnecessary suppressions of beliefs. It is carried out whenever there is a belief change, that is, whenever there is a belief update operation (this is the only possible belief change in a Neural-Logic Belief Network) and all changes associated with it. Either the general notion or the strong notion of consistency may be employed in the reasoning process. This reasoning process is a higher level procedure consists of three sub-processes: consistency checking procedures, relaxation process and consistency maintenance process. Starting from an initial belief state, it starts with a consistency check to determine which are the inconsistent beliefs caused by the belief change and relaxation is carried out to redundant suppressions. The inconsistent beliefs then go through consistency maintenance process and after that it returns to consistency check and relaxation. This process loops until the belief state is consistent and no more relaxation is possible, or all inconsistent beliefs are exhaustively suppressed. From the computational point of view, consistency reasoning process computes a final suppressed set \(CS\) and evolved set \(CE\) when a belief change takes place. These are the results of the interactions of each suppressed set \(CS_i\) and evolved set \(CE_i\) from each \(i^{th}\) consistency maintenance and relaxation sub-processes.
<table>
<thead>
<tr>
<th>Case</th>
<th>$a$</th>
<th>$b$</th>
<th>$a \land b$</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 +</td>
<td>(0, 1)</td>
<td>(0, 1)</td>
<td>(0, 1)</td>
<td>G- and S-consistent.</td>
</tr>
<tr>
<td>2 +</td>
<td>(0, 1)</td>
<td>(0, 0)</td>
<td>(0, 1)</td>
<td>G- and S-consistent.</td>
</tr>
<tr>
<td>3 +</td>
<td>(0, 1)</td>
<td>(1, 0)</td>
<td>(0, 1)</td>
<td>G- and S-consistent.</td>
</tr>
<tr>
<td>4 +</td>
<td>(0, 0)</td>
<td>(0, 1)</td>
<td>(0, 1)</td>
<td>G- and S-consistent.</td>
</tr>
<tr>
<td>5 **</td>
<td>(0, 0)</td>
<td>(0, 0)</td>
<td>(0, 1)</td>
<td>G- and S-consistent.</td>
</tr>
<tr>
<td>6 *=</td>
<td>(0, 0)</td>
<td>(1, 0)</td>
<td>(0, 1)</td>
<td>G-consistent but could be S-inconsistent.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>— under S-consistency reasoning, suppress</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>the weakest sub-expression to (0, 1).</td>
</tr>
<tr>
<td>7 +</td>
<td>(1, 0)</td>
<td>(0, 1)</td>
<td>(0, 1)</td>
<td>G- and S-consistent.</td>
</tr>
<tr>
<td>8 *=</td>
<td>(1, 0)</td>
<td>(0, 0)</td>
<td>(0, 1)</td>
<td>G-consistent but could be S-inconsistent.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>— under S-consistency reasoning, suppress</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>the weakest sub-expression to (0, 1).</td>
</tr>
<tr>
<td>9 @=</td>
<td>(1, 0)</td>
<td>(1, 0)</td>
<td>(0, 1)</td>
<td>G-inconsistent and S-inconsistent.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>— under G-consistency reasoning, suppress</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>the sub-expression(s) to (0, 0).</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>— under S-consistency reasoning, suppress</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>the weakest sub-expression to (0, 1).</td>
</tr>
<tr>
<td>10 @=</td>
<td>(0, 1)</td>
<td>(0, 1)</td>
<td>(0, 0)</td>
<td>G- and S-inconsistent — suppress all</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>sub-expressions to (0, 0).</td>
</tr>
<tr>
<td>11 @=</td>
<td>(0, 1)</td>
<td>(0, 0)</td>
<td>(0, 0)</td>
<td>G- and S-inconsistent — suppress all</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>sub-expressions to (0, 0).</td>
</tr>
<tr>
<td>12 @=</td>
<td>(0, 1)</td>
<td>(1, 0)</td>
<td>(0, 0)</td>
<td>G- and S-inconsistent — suppress all</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>sub-expressions to (0, 0).</td>
</tr>
<tr>
<td>13 @=</td>
<td>(0, 0)</td>
<td>(0, 1)</td>
<td>(0, 0)</td>
<td>G- and S-inconsistent — suppress all</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>sub-expressions to (0, 0).</td>
</tr>
<tr>
<td>14 +</td>
<td>(0, 0)</td>
<td>(0, 0)</td>
<td>(0, 0)</td>
<td>G- and S-consistent.</td>
</tr>
<tr>
<td>15 +</td>
<td>(0, 0)</td>
<td>(1, 0)</td>
<td>(0, 0)</td>
<td>G- and S-consistent.</td>
</tr>
<tr>
<td>16 @=</td>
<td>(1, 0)</td>
<td>(0, 0)</td>
<td>(0, 0)</td>
<td>G- and S-inconsistent — suppress all</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>sub-expressions to (0, 0).</td>
</tr>
<tr>
<td>17 +</td>
<td>(1, 0)</td>
<td>(0, 0)</td>
<td>(0, 0)</td>
<td>G- and S-consistent.</td>
</tr>
<tr>
<td>18 @=</td>
<td>(1, 0)</td>
<td>(1, 0)</td>
<td>(0, 0)</td>
<td>G- and S-inconsistent — suppress all</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>sub-expressions to (0, 0).</td>
</tr>
<tr>
<td>19</td>
<td>(0, 1)</td>
<td>(0, 0)</td>
<td>(1, 0)</td>
<td>not applicable</td>
</tr>
<tr>
<td>20</td>
<td>(0, 1)</td>
<td>(0, 0)</td>
<td>(1, 0)</td>
<td>not applicable</td>
</tr>
<tr>
<td>21</td>
<td>(0, 1)</td>
<td>(1, 0)</td>
<td>(1, 0)</td>
<td>not applicable</td>
</tr>
<tr>
<td>22</td>
<td>(0, 0)</td>
<td>(0, 1)</td>
<td>(1, 0)</td>
<td>not applicable</td>
</tr>
<tr>
<td>23</td>
<td>(0, 0)</td>
<td>(0, 0)</td>
<td>(1, 0)</td>
<td>not applicable</td>
</tr>
<tr>
<td>24</td>
<td>(0, 0)</td>
<td>(1, 0)</td>
<td>(1, 0)</td>
<td>not applicable</td>
</tr>
<tr>
<td>25</td>
<td>(1, 0)</td>
<td>(0, 0)</td>
<td>(1, 0)</td>
<td>not applicable</td>
</tr>
<tr>
<td>26</td>
<td>(1, 0)</td>
<td>(0, 0)</td>
<td>(1, 0)</td>
<td>not applicable</td>
</tr>
<tr>
<td>27 +</td>
<td>(1, 0)</td>
<td>(0, 0)</td>
<td>(1, 0)</td>
<td>G- and S-consistent.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>— not applicable —</td>
</tr>
</tbody>
</table>

Table 2: All possible cases for a two input AND relation $a \land b$. 
Before a belief update operation is carried out, the existing belief state $S_0$ is computed. This serves as the reference for comparing the degree-of-belief values in the TAO for logical suppressions. For each additive belief update operation (via Add, Update or Revise operator) where belief change may arise, depending on the type of CNF logical expression involved, consistency reasoning is carried out as follows:

1. If the logical expression introduced to the belief state is a simple proposition (e.g. $a, \neg b$), if it is not believed in the successive belief state after the belief update operation, which indicates that it has no impact on the belief state, then no consistency reasoning is required. Otherwise, if the new belief is accepted, then consistency check and possible logical relaxations are carried out to the belief state starting from this proposition. Consistency maintenance is then carried out to the related beliefs if there is inconsistency using the $S_0$ as the reference belief state. This consistency reasoning process of consistency check, possible logical relaxations and consistency maintenance for inconsistent beliefs is repeated until all inconsistencies are resolved and over-suppressed beliefs are relaxed, or all inconsistent beliefs are exhaustively suppressed.

2. The logical expression added to the belief state is either a pure disjunctive expression with a proposition-value of $(1, 0)$ (e.g. $a \lor b (((1, 0), \deg[a \lor b]))$, or a simple conjunctive expression with a proposition-value of $(0, 1)$ which is exactly the same as a disjunctive expression with the negation of all the conjuncts as its disjuncts (e.g. $a \land b (((0, 1), \deg[a \land b])$ is equivalent to $\neg a \lor \neg b (((1, 0), \deg[\neg a \lor \neg b]))$. The same consistency reasoning process described in (1) above is carried out.

3. If the logical expression introduced is a multiple conjunction, that is, it is either a conjunctive expression with a proposition-value of $(1, 0)$ (e.g. $(\neg a \lor c) \land (a \lor b)$), or a pure disjunctive expression with a proposition-value of $(0, 1)$ (e.g. $a \lor b (((0, 1), \deg[a \lor b])$) which is equivalent to the conjunctive expression $\neg a \land \neg b (((1, 0), \deg[\neg a \land \neg b]))$. Since no degree-of-belief value is given about each conjunct and by proposition 1, all conjuncts must have a degree-of-belief value at least as great as that of the logical expression, the most conservative assumption is to believe all the conjuncts at the same degree-of-belief as the conjunctive logical expression. This means that whenever a conjunctive expression is introduced, all the conjuncts are introduced at the same degree-of-belief value as the conjunctive expression by a full suppression. The same consistency reasoning process described above is carried out to maintain consistency of the resulting belief state.

This procedure asserts that more and more logical suppressions are introduced into the belief state by each round of the consistency maintenance process, and the process terminates when the belief state is consistent or all relevant inconsistent sub-expressions are exhaustively suppressed.

When a belief (a simple proposition or a logical expression) is deleted from the belief state via the Forget or Remove operators, or, is temporarily suppressed by a Not-Conclude operator, similar consistency reasoning process is carried out to ensure consistency of overall beliefs.
The following example shows the two types of human-like consistency reasoning process.

Example 1: Who is telling lies?
Suppose during an investigation, three persons are involved: Charles, Diana and Elizabeth. To begin with, all of them are assumed not to be telling lies. From the general impression, we believe that it is probably the case where Not-“Charles tells lies”, it is likely that Not-“Diana tells lies”, and it is usually that Not-“Elizabeth tells lies”, then we shall have the following in our base beliefs:

(1) $\neg$ Charles tells lies (probably).
(2) $\neg$ Diana tells lies (likely).
(3) $\neg$ Elizabeth tells lies (usually).

To simplify the propositions, let $a$ denotes the proposition “Charles tells lies” in (1), $b$ denotes the proposition in (2) and $c$ denotes the proposition in (3). If a piece of evidence given by Charles and Diana contradicts each other and it is strongly believed that either one of them is telling lies or both of them are telling lies, then we will have the following belief:

(4) Charles tells lies $\lor$ Diana tells lies (strongly).

Or, in the simplified form, $a \lor b$. If further evidence shows that Elizabeth’s statements support very strongly what Charles says, it is then believed very strongly that Not-“Elizabeth tells lies” implies Not-“Charles tells lies”, i.e. $\neg c \rightarrow \neg a$ or equivalently in a conjunctive normal form $\neg a \lor c$:

(5) $\neg$ Charles tells lies $\lor$ Elizabeth tells lies (very-strongly).

If the TAO for these beliefs is: probably < likely < usually < strongly < very strongly, then after five Add operations for beliefs (1) to (5), we shall have the belief state $S_1$ in the Neural-Logic Belief Network expressed by their relative positions on a TAO scale as shown in figure 5.

Under G-consistency reasoning, the consistency checking procedure reveals that $a \lor b$ is inconsistent with $\neg a$ and $\neg b$. A this corresponds to category 4 in the consistency maintenance process, a non-supporting logical suppression is introduced to the weakest inconsistent sub-expression $\neg a$:

$$\neg a \overset{\text{sup}}{\Rightarrow} (a, \text{deg}[-a])$$

and this is shown in belief state $S_2$ in the same figure. Further consistency check shows that $S_2$ is G-consistent and no relaxation is necessary. The suppressed set $CS_2$ is $\{a ((0, 1), \text{probably})\}$ and the evolved set $CE_2 = \{a ((0, 0), \text{probably})\}$. The final beliefs consist of beliefs (2) to (5) and the belief “Charles tells lies” becomes contradictory, i.e. we cannot conclude whether he tells lies or not from the present evidence. This shows that under the general notion of consistency reasoning, the intuitive notion of giving everybody the benefit of the doubt from human reasoning is preserved and changes to the belief state is minimum.
Figure 5: The consistency reasoning processes for the example: who tells lies?

Under S-consistency reasoning, the consistency checking procedure also reveals that \( a \lor b \) is inconsistent with \( \neg a \) and \( \neg b \). Since there exist a unique weakest inconsistent disjunct \( \neg a \), it is logically suppressed to \( a \) by (under category 4 of the consistency maintenance process):

\[
\neg a \overset{\lor b}{\rightarrow} (a, \text{ strongly})
\]

This results in the belief state \( S_{2-1} \) as shown in figure 5. The suppressed set \( C S_{2-1} = \{ a ((0, 1), \text{ probably}) \} \) and the evolved set \( CE_{2-1} = \{ a ((1, 0), \text{ strongly}) \} \).

The next round of logical relaxation and consistency check are carried out to \( S_{2-1} \). It shows that now \( \neg a \lor c \) is inconsistent with the new suppressed belief \( a \). Using \( S_1 \) as the base beliefs, a supporting logical suppression

\[
\neg a \overset{\lor c}{\rightarrow} (\neg a, \text{ very strongly})
\]

is created and the resulting belief state is \( S_{2-2} \) (category 4). The next round of consistency check indicates that now \( a \lor b \) is inconsistent with \( \neg a \) and \( \neg b \) again. This time, since \( \neg a \) has already been suppressed by \( a \lor b \), the only other alternative is to suppress \( \neg b \) to \( b \):

\[
\neg b \overset{\lor b}{\rightarrow} (b, \text{ strongly})
\]

This also corresponds to category 4 where there exist a unique weakest disjunct, \( \neg b \). \( a \lor b \) is now being fully-suppressed. The resulting belief state is \( S_{2-3} \) as shown in figure 5. Further consistency check shows that \( S_{2-3} \) is S-consistent and no redundant logical suppression requires relaxation. The final suppressed set for the S-consistency reasoning
The general notion of consistency reasoning.

The strong notion of consistency reasoning.

The final belief state consists of all the above beliefs at a proposition-value of (0, 0).

Figure 6: An example on S-consistency reasoning where all beliefs are fully suppressed.

CS₂ is \{a ((0, 1), probably), b ((0, 1), likely)\} and the evolved set CE₂ = \{a ((0, 1), very strongly), b ((1, 0), strongly)\}.

S-consistency reasoning requires that if possible, find the most likely belief(s) that supports the belief state as a whole. This corresponds to the notion that there must always be an explanation for a problem. In this example, the final beliefs include (3) to (5), and in addition, the belief for Not-“Charles tells lies” is strengthened by (5) to a degree-of-belief of “very strongly”. This means that in looking for a supporting element to (4), the belief “Diana tells lies” is introduced into the belief state.

In this example, if the initial reliabilities for beliefs (1) to (3) are the same, say “usually”, then under both G- and S- consistency reasoning, “Charles tells lies” and “Diana tells lies” will be suppressed to a contradictory state.

Example 2: When all beliefs contradict each other.

This is a modification of example 1.

If belief (5) now has the same degree-of-belief as (4), i.e. “strongly”, and further investigation indicates strongly that “Elizabeth tells lies” implies Not-“Diana tells lies”, i.e. c → ¬b or equivalently in a conjunctive normal form ¬c ∨ ¬b below:

(6) ¬Elizabeth tells lies ∨ ¬Diana tells lies (strongly).
Note that now (4), (5) and (6) contradict each other and the corresponding initial belief state is shown in figure 6. According to G-consistency reasoning, we just suppress the belief “Charles tells lies” to a contradictory state as shown in the same figure. But under S-consistency reasoning, every belief becomes inconsistent with each other during the reasoning process, and all beliefs are suppressed to the fullest which give rise to a belief state of contradictory beliefs as shown in figure 6.

Given that a Neural-Logic Belief Network is a representation of a finite set of beliefs and the consistency reasoning process is built on a procedure that more and more logical suppression is introduced from the logical expressions to its sub-expressions until it is consistent (except the redundant ones which does not affect the belief state by relaxing them). When consistency reasoning progresses, the more beliefs will be suppressed either to support their logical expressions or to a contradictory state, the less inconsistency is possible and the more inconsistencies are resolved, and more logical expressions will be exhaustively suppressed (include those that are fully suppressed). As the number of logical suppressions can be carried out by each logical expression is limited by the number of sub-expressions in them, there will be a time when no more suppression is possible, then consistency reasoning process will terminate finitely and since exhaustively suppressed beliefs are consistent, it will yield a consistent belief state (see proof in Appendix).

### 3.7 Mixing the Two Notions of Consistency Reasoning

In all previous sections, this technical report assumes that during consistency reasoning, either the general- or strong- notion is uniformly used. This is not necessarily the case. Human beings sometimes seek explanations to some beliefs (logical expressions) but for others, we may be contented to accept something that is minimally consistent. This means that for some logical expressions, S-consistency reasoning is more suitable but for others, G-consistency reasoning is sufficient. To achieve the mixing of these two type of consistency reasoning, we need something to differentiate different requirements at each logical expression. One simpler way is to assign a flag to each logical expression indicating whether the general- or strong- notion is to be used during consistency reasoning process, and consistency reasoning is carried out in the same way described in the previous sub-section.

### 4 Discussion

Human beings resolve contradictory beliefs by comparing the relative strength of the beliefs and suppressing the weaker ones in order to achieve consistency. Based on this philosophy, we have proposed a way of reasoning about inconsistencies in belief states represented by a three-valued Neural-Logic Belief Networks. This consistency reasoning process follows the principles of minimal change to the belief knowledge and only the weakest relevant beliefs are affected. These weaker beliefs are temporarily suppressed by
stronger beliefs and over-suppressed beliefs are relaxed during the consistency reasoning process. Using logical suppressions rather than set exclusions, no belief is lost through the consistency reasoning process.

There are two notions of human-like consistency reasoning: general and strong. Both of them are based on a three-valued formalism and they tolerate more inconsistencies than strict classical logical consistency. The general notion is more general than the strong notion. Changes to the belief states to maintain the general notion of consistency (G-consistency) is strictly minimum as when there is inconsistency, the weakest conflicting beliefs are suppressed to a contradictory state where they are neither believed nor not believed until further information is available. The strong notion of consistency reasoning (S-consistency) is similar to the general notion except that a more stringent view is taken and a particular type of modus ponens inference rule is used to derive at new beliefs when there is inconsistency. This happens when there is a unique weakest inconsistent sub-expression, and it is then logically suppressed to support its logical expression. With consistency reasoning carried out for all belief changes associated with the belief update operations, belief states will always be consistent for commonsense reasoning.

This is an attempt in reasoning about inconsistency of beliefs outside the restrictions of classical logic based knowledge representation systems. It will be interesting to investigate our notions of consistency with paraconsistency systems such as [Sl91]. We would also like to explore the notion of partial inconsistency and lazy consistency reasoning which is another step closer to more human-like reasoning. The fundamental reason for doing this is obvious: we do not always have control to the availability of resources in real-life reasoning. That is, consistency reasoning is not carried out to some belief changes immediately as it could be due to limited time or space, or lack of interests in doing so at that particular moment. This causes the beliefs to be partially inconsistent. When certain part of a inconsistent belief state is required for further reasoning and/or resources become available, we then perform lazy consistency reasoning to determine what are the current coherent beliefs. This certainly is very useful for extracting out a consistent set of beliefs from an initially inconsistent theory.

Appendix

To show that the consistency reasoning process terminates finitely and always resulting in a consistent belief state, we need the following lemmas.

Lemma 3 A fully suppressed logical expression is consistent.

Proof: For G-consistency reasoning, only non-supporting logical suppressions are introduced into the belief state. A logical expression \( \Phi \) with \( n \) sub-expressions is fully suppressed means all its sub-expressions are suppressed to a contradictory state and at least \( n-1 \) numbers of counter-suppressions are blocking the effects of the non-supporting logical suppressions from \( \Phi \). There are two possibilities:
**Case 1:** If there are \( n - 1 \) number of counter-suppressions to logical suppressions of \( \Psi \) from other stronger beliefs, then the last non-supporting logical supression is not being counter-suppressed and \( \Psi \) is still being believed in the belief state after full supression. Under the definition of G-consistency, since the last supression has effectively maintain consistency by making the last sub-expression suppressed into a non-opposing element, it is consistent.

**Case 2:** Otherwise, if after full supression, all of them have been counter-suppressed by other stronger beliefs, the negation of \( \Psi \) will be believed in the resulting belief state since the opposing beliefs of all its sub-expressions are believed. In this case, it is consistent.

Under S-consistency reasoning, full supression from a logical expression \( \Psi \) to its \( n \) sub-expressions \( \alpha_1, \alpha_2, \ldots, \alpha_n \) occur only when after each logical supression of one of the \( \alpha_i \) (where \( 0 < i \leq n \)), it is counter-suppressed by another stronger belief and \( \Psi \) becomes inconsistent again. The next round of consistency maintenance imposes another logical supression from \( \Psi \) to other of its sub-expression(s). This suppressing and counter suppressing process goes on until all sub-expressions of \( \Psi \) are suppressed. There are two possibilities for \( \Psi \) to be fully suppressed:

**Case 1:** If the last logical supression from \( \Psi \) is not counter-suppressed by other stronger beliefs in the belief state (i.e. there are only \( n - 1 \) counter-suppressions), \( \Psi \) will be believed and the effect of the last logical supression makes \( \Psi \) a consistent belief.

**Case 2:** Otherwise, \( \Psi \) has \( n \) counter-suppressions to its \( n \) logical supressions. Since all sub-expressions of \( \Psi \) are non-supporting to \( \Psi \) and have higher degree-of-belief values than \( \text{deg}[\Psi] \), \( \Psi \) will either be in its negation or in a contradictory state having a proposition-value of \( (0, 0) \). As the original value of \( \Psi \) is no longer believed and its new status is supported by other beliefs, it is consistent. \( \Box \)

**Lemma 4** An exhaustively suppressed logical expression is consistent.

**Proof:** If a logical expression \( \Psi \) with \( n \) sub-expressions is fully suppressed, from Lemma 3, it is consistent. \( \Psi \) is not being fully suppressed when it has \( k \) logical supressions \( (k < n) \) and \( n - k \) sub-expressions are not logically suppressed by \( \Psi \).

For G-consistency reasoning, when after exhaustively supressing the weaker relevant opposing \( k \) sub-expressions from \( \Psi \), all of them are counter-suppressed by other stronger beliefs to some opposing beliefs stronger than \( \Psi \) and no more supression is possible, this means that all sub-expressions of \( \Psi \) are opposing it and have higher degree-of-belief values than \( \Psi \). In this case, the negation of \( \Psi \) will be believed and it is logically consistent.

For S-consistency reasoning, when all \( k \) logical supressions from \( \Psi \) are counter-suppressed by other stronger beliefs to some non-supporting beliefs and no more supression is possible from \( \Psi \), these \( k \) counter-suppressed beliefs and all other \( n - k \) non-supressed
sub-expressions of $\Psi$ must have higher degree-of-belief values than $\Psi$ and they are either opposing to $\Psi$ or in a contradictory state with a proposition-value of $(0, 0)$. If all the sub-expressions oppose $\Psi$ then the negation of $\Psi$ will be believed and the belief state is consistent. If some of the sub-expressions are in a contradictory state, then according to the definition of S-consistency, it is consistent. □

**Lemma 5** The maximum number of logical suppressions can be imposed on a belief state is the sum of all the number of sub-expressions in each logical expression.

**Proof**: Following from proposition 2, the maximum limit of logical suppressions can be imposed by each logical expression equals to the number of its sub-expressions. If maximum logical suppression is imposed for all logical expressions in a belief state (full suppression), then the total number of logical suppressions is the sum of the number of sub-expressions from all logical expressions. □

**Lemma 6** The consistency reasoning process terminates at a consistent belief state.

**Proof**: From the procedures of consistency reasoning process, there could be two modes of termination:

1. It terminates normally. That is, it terminates when there is no more inconsistent beliefs in the belief state.

2. It terminates when inconsistent beliefs are exhaustively suppressed. Following from Lemma 4 that exhaustively suppressed beliefs are consistent, the belief state that consists of consistent beliefs and the exhaustively suppressed beliefs is consistent. □

**Theorem 7 (Termination of consistency reasoning process)**

A consistency reasoning process terminates finitely in a consistent belief state.

**Proof**: From the basic definition, a Neural-Logic Belief Network is a representation of a finite set of beliefs. With a finite number of beliefs and from Lemma 5, the consistency reasoning process terminates finitely with finite total number of possible suppressions. Following from this and Lemma 6, a consistency reasoning processes will terminate finitely in a consistent belief states. □

**Acknowledgement**

This research is supported by a grant from the Australian Research Council and in part by a CSIRO Collaborative Research Scholarship from the Division of Information Technology. The first author is supported by an Equity and Merit Scholarship from the Australian Government.
References


