Online appendix for the paper

Generating Explanations for Biomedical Queries

published in Theory and Practice of Logic Programming

ESRA ERDEM, UMUT OZTOK

Sabancı University, Orhanlı, Tuzla, İstanbul 34956, Turkey
(*e-mail:* {esraerdem,uoztok}@sabanciuniv.edu)

submitted 25 October 2012; revised 17 July 2013; accepted 19 August 2013

Appendix A Proofs

We provide proofs of the theoretical results presented in the paper: the algorithmic analysis for generating shortest explanations (Propositions 2, 3, and 4), the algorithmic analysis for generating k different explanations (Propositions 5, 6, and 8), and the analysis for the relationship between an explanation and a justification. (Propositions 10 and 11).

A.1 Generating Shortest Explanations

In Section 5 of the paper, we have analyzed three properties of Algorithm 1, namely termination, soundness and complexity, resulting in Propositions 2, 3, and 4. In the following, we show the proofs of these results.

A.1.1 Proof of Proposition 2 – Termination of Algorithm 1

To prove that Algorithm 1 terminates we need the following lemma.

Lemma 1
Given a ground ASP program Π, an answer set X for Π, an atom p in X, and a set L of atoms in X, Algorithm 2 terminates.

Proof of Lemma 1
It is sufficient to show that the recursion tree generated by Algorithm 2 is finite. That is, the branching factor of the recursion tree and the height of the recursion tree are finite. Note that each node of the recursion tree denotes a call to createTree(Π, X, d, L) for some atom or rule d and a set L of atoms.

Part (1) We show that the branching factor of the tree is finite. Branches in the tree are created in loops at Lines 5 and 13. The loop at Line 5 iterates at most the number of rules in Π and the loop at Line 13 iterates at most the number of atoms in X. As Π and X are finite, the branching factor of the tree is finite.

Part (2) We show that the height of the recursion tree is finite. Let us first make the following observation. Consider a path $\langle v_1, \ldots \rangle$ in the recursion tree. For every node v_i that denotes a call to createTree(Π, X, d_i, L_i) where d_i is an atom in X, the following holds for v_{i+2} (if exists): $L_i \subset L_{i+2}$. This follows from the two consecutive calls to createTree: L_i increases at every other call, due to Line 4. Now, assume that the height of the recursion tree is infinite. Then, there exists an infinite path $\langle v_1, \ldots \rangle$ in the recursion tree. Thus, $L_1 \subset L_3 \subset \ldots \subset L_{(2\times|X|)+1} \subset L_{(2\times|X|)+3} \subset \ldots$. Recall that we add elements into L_is only from X (Line 4). Thus, $L_{(2\times|X|)+1} = X$. Then, it is not possible to have $L_{(2\times|X|)+1} \subset L_{(2\times|X|)+3}$. As we reach a contradiction, the height of the recursion tree is finite.

□

Now, we show that Algorithm 1 terminates.

Proposition 2
Given a ground ASP program Π, an answer set X for Π, and an atom p in X, Algorithm 1 terminates.

Proof of Proposition 2
Algorithm 1 terminates only if Algorithms 2, 3, and 4 terminate. By Lemma 1, we know that Algorithm 2 terminates and that the vertex-labeled tree T returned by Algorithm 2 is finite. Since Algorithm 3 and Algorithm 4 simply traverse T (cf. Lines 2 and 6 in Algorithm 3, and Lines 5 and 10 in Algorithm 4), they also terminate. Thus, Algorithm 1 terminates.

□

A.1.2 Proof of Proposition 3 – Soundness of Algorithm 1

To show the proof of Proposition 3, we need the following necessary lemmas.

Lemma 2
Let Π be a ground ASP program, X be an answer set for Π, d be an atom in X or a rule in Π and L be a subset of X. If the vertex-labeled tree $\langle V, E, l, \Pi, X \rangle$ returned by createTree(Π, X, d, L) is not empty, then the following hold:

(i) the root of $\langle V, E \rangle$ is created in createTree(Π, X, d, L) and is mapped to d by l;
(ii) for every rule vertex $v \in V$,

$$out_E(v) = \{(v, v') \mid (v, v') \in E,\ l(v') \in B^+(l(v))\};$$

(iii) each leaf vertex is a rule vertex.

Proof of Lemma 2
Let $\langle V, E, l, \Pi, X \rangle$ be the non-empty vertex-labeled tree returned by createTree(Π, X, d, L). We show one by one that each condition in the lemma holds.

(i) Assume that d is an atom in X. Note that $d \notin L$ because otherwise $\langle V, E\rangle = \langle \emptyset, \emptyset\rangle$. Due to the call createTree(Π, X, d, L), the algorithm, at Line 3, creates a vertex v such that $l(v) = d$. We know that $E \neq \emptyset$. Then, there exists some out-going edges of v. At Line 9, the out-going edges of v are formed. Due to Line 8, each vertex v' in (v, v') is the root of a vertex-labeled tree. Moreover, there is no part of the algorithm that adds a "parent" to a vertex. Therefore, v is the root of $\langle V, E\rangle$.

Similar reasoning can be applied for the case where d is a rule in Π.

(ii) Let $v \in V$ be a rule vertex. Let $S_v = \{(v, v') \,|\, (v, v') \in E, l(v') \in B^+(l(v))\}$ denoting the set of out-going edges of v to atom vertices whose labels are in the positive body of the rule that labels v. We show that $\mathit{out}_E(v) = S_v$.

Let $(v, v') \in \mathit{out}_E(v)$. Then, $(v, v') \in E$. Edges are created at Lines 9 and 17. As v is a rule vertex, (v, v') must be created at Line 17. Then, by Line 16 and the condition at Line 13, $l(v') \in B^+(l(v))$. So, $(v, v') \in S_v$ (i.e., $\mathit{out}_E(v) \subseteq S_v$).

Let $(v, v') \in S_v$. Then, by the definition of S_v, $(v, v') \in E$. Thus, $(v, v') \in \mathit{out}_E(v)$ (i.e., $S_v \subseteq \mathit{out}_E(v)$).

(iii) Assume otherwise. Then, there exists an atom vertex $v \in V$ such that it is a leaf vertex. Vertices are created at Lines 3 and 12. As v is an atom vertex, it must be created at Line 3. Since it is a leaf vertex, condition at Line 7 never holds. But, then condition at Line 10 holds. Then, the call where v is created returns an empty set of vertices. That is, v cannot be in V.

□

Lemma 3
Let Π be a ground ASP program and X be an answer set for Π. For the two subsequent calls createTree(Π, X, d, L) and createTree(Π, X, d', L') (i.e., createTree(Π, X, d', L') being called right after createTree(Π, X, d, L)) on a path in the recursion tree for some execution of Algorithm 2, the following hold

(i) If d is an atom, then $L' = L \cup \{d\}$;
(ii) If d is a rule, then $L' = L$.

Proof of Lemma 3
Let createTree(Π, X, d, L) and createTree(Π, X, d', L') be two subsequent calls on a path in the recursion tree for some execution of Algorithm 2. We show one by one that the conditions in the lemma hold.

(i) Assume that d is an atom. Then, createTree(Π, X, d', L') is called at Line 6. Due to Line 4, $L' = L \cup \{d\}$.

(ii) Assume that d is a rule. Then, createTree(Π, X, d', L') is called at Line 13. As L is not modified prior to this call, $L' = L$.

□

Lemma 4
Let Π be a ground ASP program, X be an answer set for Π and $\langle V, E, l, \Pi, X\rangle$ be the vertex-labeled tree returned by execution Exc of Algorithm 2. Let createTree(Π, X, d, L) and createTree(Π, X, d', L') be the two subsequent calls (i.e., createTree(Π, X, d', L') being called right after createTree(Π, X, d, L)) on a path in the recursion tree for Exc, where each call on the path returns a nonempty vertex-labeled tree. Let v and v' be two vertices created by createTree(Π, X, d, L) and createTree(Π, X, d', L'), respectively, such that $l(v) = d$ and $l(v') = d'$. Then, $(v, v') \in E$.

Proof of Lemma 4
Notice that either d is an atom and d' is a rule or vice versa. We show that the lemma holds for the former case. The latter case can be shown similarly. Assume that d is an atom and d' is a rule. Then, createTree(Π, X, d', L') must be called at Line 6 within createTree(Π, X, d, L). As none of the calls on the path returns empty vertex-labeled tree, the condition at Line 7 holds in createTree(Π, X, d, L). Then, an edge (v, v') is added to E at Line 9. Note that v is created in createTree(Π, X, d, L) at Line 3 and $l(v) = d$. Also, due to Line 8, v' is the root of the vertex-labeled tree returned by createTree(Π, X, d', L'). Then, by Lemma 2, v' is a vertex created in createTree(Π, X, d', L') and $l(v') = d'$.
□

Lemma 5
Let Π be a ground ASP program, X be an answer set for Π, d be an atom in X and $T = \langle V, E, l, \Pi, X\rangle$ be the vertex-labeled tree returned by createTree($\Pi, X, d, \emptyset$). If T is not empty, then for every node createTree(Π, X, d', L') in the recursion tree for createTree($\Pi, X, d, \emptyset$), where createTree(Π, X, d', L') and its ancestors return nonempty vertex-labeled trees, $L' = anc_T(v)$ where $l(v) = d'$.

Proof of Lemma 5
Assume that T is not empty. Then, we prove the lemma by induction on the depth of a node in the recursion tree for createTree($\Pi, X, d, \emptyset$).

Base case: Note that the node at depth 0 in the recursion tree for createTree($\Pi, X, d, \emptyset$) returns a nonempty vertex-labeled tree and it has no ancestors. By Lemma 2, the root v of $\langle V, E\rangle$ is mapped to d by l, i.e., $l(v) = d = d'$. As the root of a tree does not have any ancestors, $anc_T(v) = \emptyset = L'$.

Induction step: As an induction hypothesis, assume that for every node createTree(Π, X, d', L') at depth less than n in the recursion tree for createTree($\Pi, X, d, \emptyset$), where createTree(Π, X, d', L') and its ancestors return nonempty vertex-labeled trees, $L' = anc_T(v)$ where $l(v) = d'$. Let createTree(Π, X, d', L') be a node at depth n in the recursion tree for createTree($\Pi, X, d, \emptyset$), where createTree(Π, X, d', L') and its ancestors return nonempty vertex-labeled trees. We show that $L' = anc_T(v)$ where $l(v) = d'$. For that, we need to consider two cases: d' is an atom and d' is a rule. Let createTree(Π, X, p, L_p) be the parent of createTree(Π, X, d', L'), as illustrated in Figure A 1.

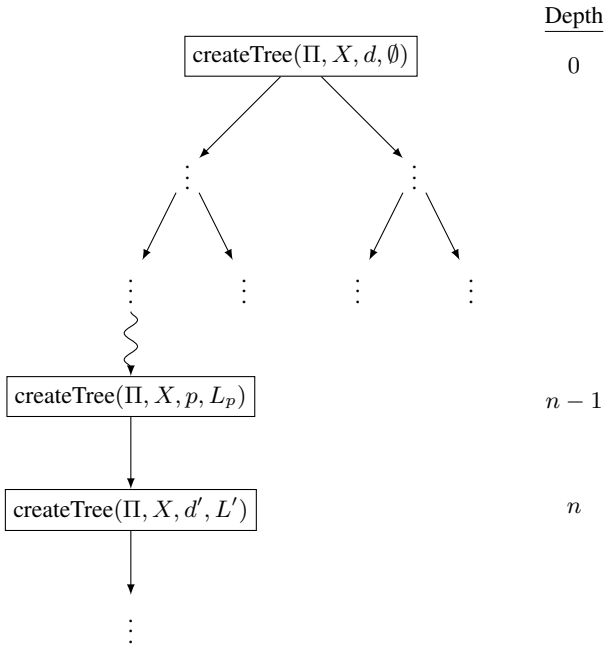

Fig. A 1: Part of the recursion tree for createTree($\Pi, X, d, \emptyset$).

Case 1. Assume that d is an atom. Then, p must be a rule. By Lemma 3, $L' = L_p$. Notice that the depth of the parent node is $n-1$. Then, by the induction hypothesis, $L_p = \mathit{anc}_T(v)$ where $l(v) = p$. Also, by Lemma 4, $(v, v') \in E$ where $l(v') = d'$. Since v is a rule vertex, $\mathit{anc}_T(v) = \mathit{anc}_T(v')$. Thus, $L' = \mathit{anc}_T(v')$ where $l(v') = d'$.

Case 2. Assume that d is a rule. Then, p must be an atom. By Lemma 3, $L' = L_p \cup \{p\}$. Notice that the depth of the parent node is $n-1$. Then, by the induction hypothesis, $L_p = \mathit{anc}_T(v)$ where $l(v) = p$. Also, by Lemma 4, $(v, v') \in E$ where $l(v') = d'$. Since v is an atom vertex, $\mathit{anc}_T(v) \cup \{l(v)\} = \mathit{anc}_T(v')$. Thus, $L' = \mathit{anc}_T(v')$ where $l(v') = d'$.

□

Proposition 12 *(Soundness of Algorithm 2)*
Let Π be a ground ASP program, X be an answer set for Π, d be an atom in X and $T = \langle V, E, l, \Pi, X\rangle$ be the vertex-labeled tree returned by createTree($\Pi, X, d, \emptyset$). If T is not empty, then T is the and-or explanation tree for d with respect to Π and X.

Proof of Proposition 12
Suppose that T is not empty. We want to show that T is the and-or explanation tree for d with respect to Π and X. For that, T must satisfy conditions $(i) - (iv)$ in Definition 2. As T is not empty, conditions $(i), (iii)$ and (iv) hold due to Lemma 2. To complete the proof, we show condition (ii) also holds in the sequel.

Let $S_v = \{(v, v') \mid (v, v') \in E,\ l(v') \in \Pi_{X, \mathit{anc}_T(v')}(l(v))\}$. Our goal is to show that for every atom vertex $v \in V$, $\mathit{out}_E(v) = S_v$. To do so, we show that $\mathit{out}_E(v) \subseteq S_v$ and that $S_v \subseteq \mathit{out}_E(v)$.

Let v be an atom vertex in V. Now, we show that $\mathit{out}_E(v) \subseteq S$. Take an arbitrary element $(v, v') \in \mathit{out}_E(v)$. Then, $(v, v') \in E$. Throughout the algorithm edges are created at Lines 9 and 17. Since v is an atom vertex, (v, v') must be created at Line 9. Then, due to the condition at Line 5 and Lemma 5, $l(v') \in \Pi_{X,\mathit{anc}_E(v')}(l(v))$. Thus, $(v, v') \in S_v$. As the last step, we show that $S_v \subseteq \mathit{out}_E(v)$. Let $(v, v') \in S_v$ be an arbitrary element. Then, by the definition of S_v, $(v, v') \in E$. Thus, trivially, $(v, v') \in \mathit{out}_E(v)$.

□

Lemma 6

Let Π be a ground ASP program, X be an answer set for Π, d be an atom in X, $T = \langle V_T, E_T, l, \Pi, X\rangle$ be the and-or explanation tree for d with respect to Π and X, v be the root of T and $\langle V, E, l, \Pi, X\rangle$ be the vertex-labeled tree returned by

$$\text{extractExp}(\Pi, X, V_T, l, v, E_T, W_T, \emptyset, \mathit{min}).$$

Then, for each $v' \in V$, we have

$$W_T(v') \leq \mathit{min}\{W_T(s) \mid s \in \mathit{sibling}_{E_T}(v')\}.$$

Proof of Lemma 6

Let $v' \in V$. Vertices are added to V at Line 8. Then, due to Line 7, v' is a rule vertex. Also, each vertex added to V corresponds to the 5^{th} parameter of the algorithm. Note that recursive calls are made at Lines 5 and 10. Since the 5^{th} parameter is a rule vertex only in the call at Line 5, the call where v' is added to V must be initiated at Line 5. Then, due to Line 3, $W_T(v') = \mathit{min}\{W_T(s) \mid s \in \mathit{sibling}_{E_T}(v')\}$.

□

Lemma 7

Let Π be a ground ASP program, X be an answer set for Π, d be an atom in X, $T = \langle V_T, E_T, l, \Pi, X\rangle$ be the and-or explanation tree for d with respect to Π and X, v be the root of T, and $\langle V, E, l, \Pi, X\rangle$ be the vertex-labeled tree returned by

$$\text{extractExp}(\Pi, X, V_T, l, v, E_T, W_T, \emptyset, \mathit{min}).$$

Let $v_1, v_2 \in V$. If $(v_1, v), (v, v_2) \in E_T$ for some $v \in V_T$, then $(v_1, v_2) \in E$.

Proof of Lemma 7

Since $v_1 \in V$, it is added to V at Line 8. Then, for each child of v_1, the algorithm is recursively called at Line 10. As $(v_1, v) \in E_T$, we make a call

$$\text{extractExp}(\Pi, X, V_T, l, v, E_T, W_T, v_1, \mathit{min}).$$

Inside that call, we add (v_1, c) to E (at Line 4) where c is a minimum weighted child of v (due to Line 3). As $(v, v_2) \in E_T$ and v_2 is a minimum weighted child of v due to Lemma 6, c is equal to v_2. Thus, $(v_1, v_2) \in E$.

□

Lemma 8
Let Π be a ground ASP program, X be an answer set for Π, d be an atom in X, $T = \langle V_T, E_T, l, \Pi, X\rangle$ be the and-or explanation tree for d with respect to Π and X, and *op* be a string *min*. Then, Algorithm 4 returns an explanation $\langle V, E, l, \Pi, X\rangle$ for d with respect to Π and X.

Proof of Lemma 8
Let v be the root of T and $S = \langle V, E, l, \Pi, X\rangle$ be the output of

$$\text{extractExp}(\Pi, X, V_T, l, v, E_T, W_T, \emptyset, \textit{min}).$$

To prove that S is an explanation for d with respect to Π and X, we need to show that there exists an explanation tree $\langle V', E', l, \Pi, X\rangle$ in T which satisfies Conditions (i) and (ii) in Definition 4 of the paper. That is, the following hold.

(i) $V = \{v \mid v \text{ is a rule vertex in } V'\}$;
(ii) $E = \{(v_1, v_2) \mid (v_1, v), (v, v_2) \in E' \text{ for some atom vertex } v \in V'\}$.

To do so, we construct a vertex-labeled tree and show that it is an explanation tree in T which satisfies above conditions. Thus, let us define a vertex-labeled tree $T' = \langle V', E', l, \Pi, X\rangle$ where

$$V' = V \cup \{v \mid v \in V_T \text{ s.t. } (v, v') \in E_T \text{ for some } v' \in V\} \tag{A1}$$
$$E' = \{(v, v') \mid v, v' \in V' \text{ s.t. } (v, v') \in E_T\} \tag{A2}$$

We now show that T' is an explanation tree in T, i.e., T' satisfies Conditions (i)–(iv) in Definition 3.

(i) Due to Line 8 and that $v \in V_T$, $V \subseteq V_T$. So, by (A1), $V' \subseteq V_T$. Also, by (A2), $E' \subseteq E_T$.

(ii) In the first call of Algorithm 4, v is the root of T. Then, at Line 5, the algorithm is called with a child c of v. Note that c is a rule vertex. Due to Line 8, for some $v' \in V$, $(v, v') \in E_T$.

(iii) Let v' be an atom vertex in V'. Then, by (A1) and (A2), $(v', v'') \in E'$ for some $v'' \in V$. This ensures that $\textit{deg}'_E(v') \geq 1$. Assume that $\textit{deg}'_E(v') > 1$. Then, for some $v''' \in V'$ ($v'' \neq v'''$), $(v', v''') \in E'$. Then, $v''' \in V$. This is not possible due to Line 3. Thus, $\textit{deg}_E(v') = 1$.

(iv) Let v' be a rule vertex in V'. Then, by (A1) $v' \in V$ and v is added into V at Line 8. Due to Line 9 and (A1), every child c of v is in V'. Then, by (A2), $(v, c) \in E'$. That is, $\textit{out}_{E_T}(v) \subseteq E'$.

As a last step, we show that T' satisfies Conditions (i) and (ii) in Definition 4.

(i) Note that every element in the set $\{v \mid v \in V_T \text{ s.t. } (v, v') \in E_T \text{ for some } v' \in V\}$ is an atom vertex. Then, by (A1), $V = \{v \mid v \text{ is a rule vertex in } V'\}$;

(ii) Let $S = \{(v_1, v_2) \mid (v_1, v), (v, v_2) \in E' \text{ for some atom vertex } v \in V'\}$. We show that $E = S$.

Let $(v_1, v_2) \in E$. Then, (v_1, v_2) is added into E at Line 4. So, due to Lines 2 and 3, we know that there exists an atom vertex $v \in V_T$ such that $(v, v_2) \in E_T$. Then, by (A1), $v \in V'$ and, by (A2), $(v, v_2) \in E'$. Also, by Line 9, we know that $(v_1, v) \in E_T$. Then, by (A2), $(v_1, v) \in E'$. Thus, $(v_1, v_2) \in S$. That is, $E \subseteq S$.

Let $(v_1, v_2) \in S$. Then, for some atom vertex $v \in V'$, $(v_1, v), (v, v_2) \in E'$. By (A2), $v_1, v_2 \in V$ and $(v_1, v), (v, v_2) \in E_T$. Then, due to Lemma 7, $(v_1, v_2) \in E$. That is, $S \subseteq E$.

□

Lemma 9

Let Π be a ground ASP program, X be an answer set for Π, and p be an atom in X. Let $T = \langle V, E, l, \Pi, X\rangle$ be the and-or explanation tree for p with respect to Π and X, $T' = \langle V', E', l, \Pi, X\rangle$ be an explanation tree in T and v be a rule vertex in V'. Then, the following inequality holds for the weight of v

$$W_T(v) \leq 1 + |\{u' \mid u' \in \mathit{des}_{T'}(v)\}|. \tag{A3}$$

Proof of Lemma 9

We prove the lemma by induction on the height of a rule vertex in the explanation tree.

Base case: Let u be a rule vertex in V' at height 0. Then, u is a leaf vertex. By the definition of the weight function, $W_T(u) = 1$. Then, (A3) holds.

Induction step: As an induction hypothesis, assume that for every rule vertex $i \in V'$ at height less than n, (A3) holds. We show that (A3) holds for every rule vertex $w \in V'$ at height $n + 1$. Let w be a rule vertex at height $n + 1$. Then, by the definition of the weight function, $W_T(w) = 1 + \sum_{w' \in \mathit{child}_E(w)} W_T(w')$. Let w' be a child of w. Note that w' is an atom vertex. By the definition of an explanation tree, $w' \in V'$ and w' has exactly one child $w'' \in V'$ which is a rule vertex. Then, by the definition of the weight function, $W_T(w') = \mathit{min}\{W_T(c) \mid c \in \mathit{child}_E(w')\}$ and thus $W_T(w') \leq W_T(w'')$. On the other hand, as the height of w'' is $n - 1$, by the induction hypothesis, $W_T(w'') \leq 1 + |\{d \mid d \in \mathit{des}_{T'}(w'')\}|$. Since w' has exactly one child $w'' \in V'$, we have

$$1 + |\{d \mid d \in \mathit{des}_{T'}(w'')\}| = |\{u \mid u \in \mathit{des}_{T'}(w')\}|.$$

That is, $W_T(w') \leq |\{u \mid u \in \mathit{des}_{T'}(w')\}|$. Then, we have

$$\begin{aligned} W_T(w) &= 1 + \textstyle\sum_{w' \in \mathit{child}_E(w)} W_T(w') \\ &\leq 1 + \textstyle\sum_{w' \in \mathit{child}_E(w)} |\{u \mid u \in \mathit{des}_{T'}(w')\}| \\ &= 1 + |\{u' \mid u' \in \mathit{des}_{T'}(w)\}|. \end{aligned}$$

□

Lemma 10
Let Π be a ground ASP program, X be an answer set for Π, p be an atom in X, T be the and-or explanation tree (with edges E) for p with respect to Π and X, v be the root of T, and T' be an explanation tree (with vertices V') in T. Then,

$$W_T(v) \leq |\{u \mid u \text{ is a rule vertex in } V'\}|.$$

Proof of Lemma 10
We want to show that the weight of v, $W_T(v)$, is at most the number of rule vertices in V'. Note that v is the root of T' and there exists exactly one vertex $v' \in V'$ such that $v' \in \mathit{child}_E(v)$ (due to Definition 3). Then, we have

$$\begin{aligned}
W_T(v) = &\quad \mathit{min}\{W_T(c) \mid c \in \mathit{child}_E(v)\} && \text{(by Definition 6)}\\
\leq &\quad W_T(v') && \text{(as } v' \in \mathit{child}_E(v)\text{)}\\
\leq &\quad 1 + |\{v'' \mid v'' \in \mathit{des}_{T'}(v')\}| && \text{(by Lemma 9)}\\
= &\quad |\{u \mid u \text{ is a rule vertex in } V'\}|. && \text{(as } v' \text{ is the only child of } v \text{ in } T'\text{)}
\end{aligned}$$

□

Lemma 11
Let Π be a ground ASP program, X be an answer set for Π, p be an atom in X, T be the and-or explanation tree for p with respect to Π and X, v be the root of T, and $\langle V, E, l, \Pi, X\rangle$ be an explanation for p with respect to Π and X. Then, $W_T(v) \leq |V|$.

Proof of Lemma 11
We show that the weight of v, $W_T(v)$, is at most $|V|$. By the definition of an explanation, there exists an explanation tree T' (with vertices V') of T such that $|V| = |\{v' \mid v' \text{ is a rule vertex in } V'\}|$. By Lemma 10, $W_T(v) \leq |\{v' \mid v' \text{ is a rule vertex in } V'\}|$. Thus, $W_T(v) \leq |V|$.

□

Lemma 12
Let Π be a ground ASP program, X be an answer set for Π, p be an atom in X, $T = \langle V, E, l, \Pi, X\rangle$ be the and-or explanation tree for p with respect to Π and X, and $T' = \langle V', E', l, \Pi, X\rangle$ be an explanation tree in T. If we have

$$W_T(v) \leq \mathit{min}\{W_T(s) \mid s \in \mathit{sibling}_E(v)\}. \tag{A4}$$

for every rule vertex $v \in V'$, then the following holds for every rule vertex $v \in V'$.

$$W_T(v) = 1 + |\{u' \mid u' \in \mathit{des}_{T'}(v)\}|. \tag{A5}$$

Proof of Lemma 12
Assume that (A4) holds for every rule vertex $v \in V'$. Then, we prove the lemma by induction on the height of a rule vertex in the explanation tree.

Base case: Let u be a rule vertex in V' at height 0. Then, u is a leaf vertex. By the definition of the weight function, $W_T(u) = 1$. As u has no descendants, (A5) holds.

Induction step: As an induction hypothesis, assume that for every rule vertex $i \in V'$ at height less than n, (A5) holds. We show that (A5) holds for every rule vertex $w \in V'$ at height $n+1$. Let $w \in V'$ be a rule vertex at height $n+1$. Then, by the definition of the weight function, $W_T(w) = 1 + \sum_{w' \in \mathit{child}_E(w)} W_T(w')$. Let w' be a child of w. Note that w' is an atom vertex. By the definition of an explanation tree, $w' \in V'$ and w' has exactly one child $w'' \in V'$, which is a rule vertex. Then, by the definition of the weight function, $W_T(w') = \mathit{min}\{W_T(c) \mid c \in \mathit{child}_E(w')\}$. Also, by (A4), $W_T(w'') \leq \mathit{min}\{W_T(s) \mid s \in \mathit{sibling}_E(w'')\}$. Then, $W_T(w'') = \mathit{min}\{W_T(c) \mid c \in \mathit{child}_E(w')\}$. Thus, $W_T(w') = W_T(w'')$. As the height of w'' is $n-1$, by the induction hypothesis, $W_T(w'') = 1+|\{u \mid u \in \mathit{des}_{T'}(w'')\}|$. As w'' is the only child of w', $W_T(w') = |\{u \mid u \in \mathit{des}_{T'}(w')\}|$. Then, we have

$$\begin{aligned} W_T(w) = & \quad 1 + \textstyle\sum_{w' \in \mathit{child}_E(w)} W_T(w') \\ = & \quad 1 + \textstyle\sum_{w' \in \mathit{child}_E(w)} |\{u \mid u \in \mathit{des}_{T'}(w')\}| \\ = & \quad 1 + |\{u' \mid u' \in \mathit{des}_{T'}(w)\}|. \end{aligned}$$

□

Lemma 13
Let Π be a ground ASP program, X be an answer set for Π, p be an atom in X, T be the and-or explanation tree (with edges E) for p with respect to Π and X, v be the root of T, and T' be an explanation tree (with vertices V') in T. If we have

$$W_T(v') \leq \mathit{min}\{W_T(s) \mid s \in \mathit{sibling}_E(v')\} \tag{A6}$$

for every rule vertex $v' \in V'$, then, the following holds

$$W_T(v) = |\{u \mid u \text{ is a rule vertex in } V'\}|.$$

Proof of Lemma 13
Assume that (A4) holds for every rule vertex $v' \in V'$. Then, we want to show that the weight of v, $W_T(v)$, is equal to the number of rule vertices in V'. Note that v is the root of T' and there exists exactly one vertex $v' \in V'$ such that $v' \in \mathit{child}_E(v)$ (due to Definition 3). Then, we have

$$\begin{aligned} W_T(v) = & \quad \mathit{min}\{W_T(c) \mid c \in \mathit{child}_E(v)\} && \text{(by Definition 6)} \\ = & \quad W_T(v') && \text{(by (A6))} \\ = & \quad 1 + |\{u' \mid u' \in \mathit{des}_{T'}(v')\}| && \text{(by Lemma 12)} \\ = & \quad |\{u \mid u \text{ is a rule vertex in } V'\}|. && \text{(as } v' \text{ is the only child of } v \text{ in } T') \end{aligned}$$

□

We are now ready to prove Proposition 3.

Proposition 3
Given a ground ASP program Π, an answer set X for Π, and an atom p in X, Algorithm 1 either finds a shortest explanation for p with respect to Π and X or returns an empty vertex-labeled tree.

Proof of Proposition 3

Algorithm 1 has two return statements; Lines 6 and 8. At Line 8, it returns an empty vertex-labeled tree. We show that what Algorithm 1 returns at Line 6, $S = \langle V', E', l, \Pi, X \rangle$, is an explanation for p with respect to Π and X and that there is no other explanation for p with respect to Π and X, with vertices V'', such that $|V''| < |V'|$.

Due to the condition at Line 2, the vertex-labeled tree found at Line 1 is not empty. Then, by Proposition 12, we know that T is the and-or explanation tree for p with respect to Π and X. Then, by Lemma 8, we know that $\langle V', E', l, \Pi, X \rangle$ found at Line 5 is an explanation for p with respect to Π and X.

Now, suppose that there exists another explanation for p with respect to Π and X, with vertices V'', such that $|V''| < |V'|$. Let v be the root of T. Then, by Lemma 11, $W_T(v) \leq |V''|$. Also, since S is an explanation for p with respect to Π and X, there exists an explanation tree with vertices V''' in T such that $V' = \{u \mid u \text{ is a rule vertex in } V'''\}$. Due to Lemma 6, $W_T(v') \leq \mathit{min}\{W_T(s) \mid s \in \mathit{sibling}_E(v')\}$ for each $v' \in V'$. Then, by Lemma 13, $W_T(v) = |\{u \mid u \text{ is a rule vertex in } V'''\}|$. This implies that $|V'| \leq |V''|$. Since this is a contradiction, S is a shortest explanation for p with respect to Π and X.

□

A.1.3 Proof of Proposition 4 – Complexity of Algorithm 1

We prove Proposition 4 which shows that the time complexity of Algorithm 1 is exponential in the size of the given answer set.

Proposition 4
Given a ground ASP program Π, an answer set X for Π, and an atom p in X, the time complexity of Algorithm 1 is $O(|\Pi|^{|X|} \times |\mathcal{B}_\Pi|)$.

Proof of Proposition 4

In Algorithm 1, all the lines, expect 1, 4 and 5, take constant amount of time. At Lines 1, 4 and 5, three different algorithms are called. At Line 1, Algorithm 2 is called. This algorithm creates the and-or explanation tree recursively. As shown in Proof of Lemma 1, the branching factor of a vertex in the recursion tree is $O(\mathit{max}\{|X|, |\Pi|\})$ and the height of the tree is $O(|X|)$. Also, at Line 5, for an atom $d \in X$, we check whether a rule in Π supports d in $O(|\mathcal{B}_\Pi|)$. Thus, the time complexity of Algorithm 2, in the worst case, is $O(\mathit{max}\{|X|, |\Pi|\}^{|X|} \times |\mathcal{B}_\Pi|)$. As $|X| \leq |\Pi|$, it is $O(|\Pi|^{|X|} \times |\mathcal{B}_\Pi|)$. At Lines 4 and 5, Algorithm 3 and Algorithm 4 are called, respectively. Algorithm 3 and Algorithm 4 simply traverse the tree T created by Algorithm 2 (cf. Lines 2 and 6 in Algorithm 3, and Lines 5 and 10 in Algorithm 4. By Proposition 12, we know that T is the and-or explanation tree for p with respect to Π and X. Since the height of T is $O(|X|)$ and the branching factor of a vertex in T is $\mathit{max}\{|X|, |\Pi|\}$, the time complexity of Algorithm 2 dominates the time complexities of others. Thus, the time complexity of Algorithm 1, in the worst case, is $O(|\Pi|^{|X|} \times |\mathcal{B}_\Pi|)$.

□

A.2 Generating k Different Explanations

In Section 6 of the paper, we have first analyzed three properties of Algorithm 5, namely termination, soundness and complexity, resulting in Propositions 5, 6, and 8. Then, we show some characteristics of its output, resulting in Proposition 7, Corollary 1 and Corollary 2. In the following, we show the proofs of these results.

A.2.1 Proof of Proposition 5 – Termination of Algorithm 5

We now prove Proposition 5 which shows that Algorithm 5 terminates.

Proposition 5
Given a ground ASP program Π, an answer set X for Π, an atom p in X, and a positive integer k, Algorithm 5 terminates.

Proof of Proposition 5
Algorithm 5 calls Algorithm 2 at Line 2. By Lemma 1, we know that Algorithm 2 terminates. Then, to show that Algorithm 5 terminates, we need to show that the loop between Lines 4–10 terminates. Due to Line 4, the loop iterates at most k times. If it iterates less than k times, it means that it is terminated at Line 6. Thus, assume that it iterates k times. Then, it is enough to show that every iteration of the loop terminates. Consider the i^{th} ($1 \leq i \leq k$) iteration of the loop. First, at Line 5, Algorithm 6 is called. Observe that this algorithm simply traverses the and-or explanation tree T created at Line 2 (cf. Lines 2, 3, 8 and 9 in Algorithm 6). Since T is finite, Algorithm 6 terminates. Next, Algorithm 4 is called at Line 7. Similar to Algorithm 6, this algorithm also simply traverses a portion of T (cf. Lines 3, 5, 9 and 10 in Algorithm 4). Hence, Algorithm 4 also terminates. As the rest of the loop consists of some assignment statements, the i^{th} ($1 \leq i \leq k$) iteration of the loop terminates. Since every iteration of the loop terminates, Algorithm 5 terminates.
□

A.2.2 Proof of Proposition 6 – Soundness of Algorithm 5

Before showing the soundness property of Algorithm 5, we provide some necessary lemmas.

Proposition 13 *(Completeness of Algorithm 2)*
Let Π be a ground ASP program, X be an answer set for Π, p be an atom in X. Let T be the and-or explanation tree p with respect to Π and X. Then, createTree$(\Pi, X, p, \{\})$ returns T.

Proof of Proposition 13
Let $\langle V, E, l, \Pi, X\rangle$ be the output of Algorithm 2. By Condition (i) in Definition 2, the root of T is an atom vertex with label p. Since p is in X and $L = \emptyset$ at the beginning of Algorithm 2, a vertex v with label p is created at Line 3 and added into V at Line 4.

Take an atom vertex $v \in V$. Then, there should be an out-going edge (v, v') of v such that $l(v') \in \Pi_{X,anc_T(v')}(l(v))$ (due to Condition (ii) in Definition 2). Note that, the set L in Algorithm 2 is essentially $anc_T(v')$, and $l(v') \in \Pi_{X,L}(l(v))$ is checked at Line 5.

Moreover, we need to ensure that v' is a rule vertex. This condition is checked at Line 6 by recursive calls.

Take a rule vertex $v \in V$. Then, for every atom a in $B^+(l(v))$, there should be an out-going edge (v, v') of v such that $l(v') = a$ (due to Condition (iii) in Definition 2). This is satisfied by the condition at Line 13. Moreover, we need to make sure that v' is atom vertex. For that, there is a recursive call at Line 14.

Take a leaf vertex $v \in V$. Then, v must be a rule vertex (due to Condition (iv) in Definition 2). Assume that v is an atom vertex. Thus, no out-going edge is defined for v at Line 9. Then, when the loop at Line 5 terminates, the condition at Line 10 is satisfied. This implies that v is not in V. As it is a contradiction, v must be a rule vertex.

□

By this proposition and Proposition 1, we know that Algorithm 2 returns the and-or explanation tree. Thus, at Line 1 of Algorithm 5, the and-or explanation tree for p with respect to Π and X is created. We prove our statements by keeping this in mind.

Lemma 14
Let Π be a ground ASP program, X be an answer set for Π, p be an atom in X, and k be a positive integer. Let n be the number of different explanations for p with respect to Π and X. Then, for the root v_r of T, at each iteration i ($1 \leq i \leq \mathit{min}\{n, k\}$) of the loop in Algorithm 5, $W_{T,R_{i-1}}(v_r) = |\mathit{RVertices}(K_i) \backslash R_{i-1}|$.

Proof of Lemma 14
Consider the i^{th} ($1 \leq i \leq \mathit{min}\{n, k\}$) iteration of the loop. At Line 5, we call Algorithm 6 and calculate $W_{T,R_{i-1}}$ for every vertex v in V. According to Algorithm 6, for an atom vertex $v \in V$, due to Line 4, $W_{T,R_{i-1}}(v) = \mathit{max}\{W_{T,R_{i-1}}(v') \mid v' \in \mathit{child}_E(v)\}$ and for a rule vertex u, due to Lines 6–9, $W_{T,R_{i-1}}(u) = x_u + \sum_{u' \in \mathit{child}_E(u)} W_{T,R_{i-1}}(u')$ where $x_u = 1$ if $u \notin R_{i-1}$, $x_u = 0$ otherwise. Let v be an atom vertex in V. Let $\{v'_1, \ldots, v'_{v_z}\}$ be a set of rule vertices that "contribute" to $W_{T,R_{i-1}}(v)$, i.e., rule vertices that appear in the expanded formula of $W_{T,R_{i-1}}(v)$. This implies that $W_{T,R_{i-1}} = x_{v'_1} + \ldots + x_{v'_{v_z}}$ where $x_{v'_j} = 1$ if $v'_j \notin R_{i-1}$, $x_{v'_j} = 0$ otherwise, for $1 \leq j \leq v_z$. Also, at Line 7 in Algorithm 5, we extract an explanation using Algorithm 4. Then, at Line 8, we assign the output $\langle V', E', l, \Pi, X\rangle$ of Algorithm 4 to K_i. Let $\mathit{RVertices}(K_i) = \{v_1, \ldots, v_z\}$. Observe in Algorithm 4 that for every atom vertex, we process its maximum weighted child recursively (Line 3), and for every rule vertex we choose every child of it recursively (Line 9). Then, due to the observation and the calculation of $W_{T,R_{i-1}}$, $\mathit{RVertices}(K_i)$ is a set of rule vertices that contribute to $W_{T,R_{i-1}}(v_r)$. Then, $W_{T,R_{i-1}}(v_r) = x_1 + \ldots + x_z$ where $x_j = 1$ if $v_j \notin R_{i-1}$, $x_j = 0$ otherwise, for $1 \leq j \leq z$. That is, $W_{T,R_{i-1}}(v_r) = |\mathit{RVertices}(K_i) \backslash R_{i-1}|$.

□

Now, we prove Proposition 6.

Proposition 6
Let Π be a ground ASP program, X be an answer set for Π, p be an atom in X, and k be a positive integer. Let n be the number of different explanations for p with respect to Π and X. Then, Algorithm 5 returns $\mathit{min}\{n, k\}$ different explanations for p with respect to Π and X.

Proof of Proposition 6

First, assume that $n \geq k$. Then, we show that Algorithm 5 returns at Line 11 a set K of k different explanations for p with respect to Π and X. Note that an element is added into K at each iteration of the loop between Lines 4–10. Since that loop iterates k times, we need to show that two properties hold: *(i)* for every iteration i ($1 \leq i \leq k$), K_i (the element added into K at the i^{th} iteration) is an explanation for p with respect to Π and X. *(ii)* for all iterations i, j ($1 \leq i < j \leq k$), K_i and K_j are different. In the following, we consider those two properties.

(i) For every iteration i ($1 \leq i \leq k$), K_i is formed at Line 8. According to Line 7, K_i is an output of Algorithm 4. Then, due to Lemma 8, we conclude that K_i is an explanation for p with respect to Π and X.

(ii) Assume otherwise. Then, there exists i, j ($1 \leq i < j \leq k$) such that $K_i = K_j$. Consider the j^{th} iteration of the loop. At Line 5, we call Algorithm 6 and calculate $W_{T,R_{j-1}}$ for every vertex v in V. Then, at Line 7, we extract an explanation using Algorithm 4. Let $K_j = \langle V', E', l, \Pi, X \rangle$, v_r be the root of $\langle V', E' \rangle$ and *RVertices*$(K_j) = \{v_1, \ldots, v_l\}$. Then, by Lemma 14, $W_{T,R_{j-1}}(v_r) = x_1 + \ldots + x_l$ where $x_z = 1$ if $v_z \notin R_{j-1}$, $x_z = 0$ otherwise, for $1 \leq z \leq l$. Since $K_i = K_j$, $\{v_1, \ldots, v_l\} =$ *RVertices*(K_i). Due to Line 10, we know that $RVertices(K_i) \subseteq R_{j-1}$. Then, for $1 \leq z \leq l$, $v_z \in R_{j-1}$. Thus, $W_{T,R_{j-1}}(v_r) = 0$. This implies that for every vertex $v \in V$, $W_{T,R_{j-1}}(v) = 0$. That is, every rule vertex in V is also in R_{j-1}. However, as we are at the j^{th} iteration and at each iteration one explanation is computed, R_{j-1} might contain rule vertices for at most $j-1$ explanations. We know that $n \geq k$. Thus, for some $v' \in V$, the following should hold: $v' \notin R_{j-1}$. As we reach a contradiction, $K_i \neq K_j$.

Now, assume that $n < k$. Then, we show that Algorithm 5 returns at Line 6 a set K of n different explanations for p with respect to Π and X. Note that at the end of n^{th} iteration of the loop, K contains n different explanations (due to the same reasoning above). Then, in the next iteration, $W_{T,R_{n+1}}(v) = 0$. This is because R_n contains the rule vertices of n different explanations and the total number of explanations for p with respect to Π and X is n. Thus, the condition at Line 6 is satisfied and n different explanations are returned.

□

A.2.3 Some Properties of Algorithm 5

Before presenting some characteristics of the output of Algorithm 5, we provide some necessary definitions and lemmas.

Definition 18 ($W_{T,R}$)
Let Π be a ground ASP program, X be an answer set for Π, p be an atom in X, $T = \langle V, E, l, \Pi, X \rangle$ be the and-or explanation tree for p with respect to Π and X, and R be a set of rule vertices in V. Then, $W_{T,R}$ is a function that maps vertices in V to nonnegative

integers as follows.

$$W_{T,R}(v) = \begin{cases} \mathit{max}\{W_{T,R}(v') \mid v' \in \mathit{child}_E(v)\} & \text{if } v \text{ is an atom vertex;} \\ \sum_{v' \in \mathit{child}_E(v)} W_{T,R}(v') & \text{if } v \text{ is a rule vertex and } v \in R; \\ 1 + \sum_{v' \in \mathit{child}_E(v)} W_{T,R}(v') & \text{otherwise.} \end{cases}$$

Lemma 15
Let Π be a ground ASP program, X be an answer set for Π, and p be an atom in X. Let $T = \langle V, E, l, \Pi, X\rangle$ be the and-or explanation tree for p with respect to Π and X, $T' = \langle V', E', l, \Pi, X\rangle$ be an explanation tree in T and R be a set of rule vertices in V. Then, for every rule vertex v in V', the following holds

$$W_{T,R}(v) \geq \begin{cases} 1 + |\{u' \mid u' \in (\mathit{des}_{T'}(v) \backslash R)\}| & \text{if } v \notin R; \\ |\{u' \mid u' \in (\mathit{des}_{T'}(v) \backslash R)\}| & \text{otherwise.} \end{cases} \tag{A7}$$

Proof of Lemma 15
We prove the lemma by induction on the height of a rule vertex in the explanation tree.

Base case: Let v be a rule vertex in V' at height 0. Then, v is a leaf vertex. By Definition 18, $W_{T,R}(v) = x_v$ where $x_v = 1$ if $v \notin R$, $x_v = 0$ otherwise. Then, (A7) holds.

Induction step: As an induction hypothesis, assume that for every rule vertex $i \in V'$ at height less than $n+1$, (A7) holds. We show that (A7) holds for every rule vertex $v \in V'$ at height $n+1$. Let v be a rule vertex in V' at height $n+1$. Then, by Definition 18, $W_{T,R}(v) = x_v + \sum_{v' \in \mathit{child}_E(v)} W_{T,R}(v')$ where $x_v = 1$ if $v \notin R$, $x_v = 0$ otherwise. Let v' be a child of v. Note that v' is an atom vertex. By Conditions (iii) and (iv) in Definition 3, $v' \in V'$ and v' has exactly one child $v'' \in V'$ which is a rule vertex. Then, by Definition 18, $W_{T,R}(v') = \mathit{max}\{W_{T,R}(c) \mid c \in \mathit{child}_E(v')\}$ and thus $W_{T,R}(v') \geq W_{T,R}(v'')$. On the other hand, as the height of v'' is $n-1$, by the induction hypothesis, (A7) holds for v''. Thus, we obtain the following.

$$W_{T,R}(v') \geq \begin{cases} 1 + |\{u' \mid u' \in (\mathit{des}_{T'}(v'') \backslash R)\}| & \text{if } v'' \notin R; \\ |\{u' \mid u' \in (\mathit{des}_{T'}(v'') \backslash R)\}| & \text{otherwise.} \end{cases} \tag{A8}$$

Since v' has exactly one child $v'' \in V'$, we derive

$$\{v''\} \cup \{d \mid d \in \mathit{des}_{T'}(v'')\} = |\{u \mid u \in \mathit{des}_{T'}(v')\}|. \tag{A9}$$

Then, due to (A8) and (A9),

$$W_{T,R}(v') \geq |\{u \mid u \in (\mathit{des}_{T'}(v') \backslash R)\}|. \tag{A10}$$

Then, to conclude the proof, we consider two cases.

Case 1. Suppose that $v \notin R$. Then, we can derive the following.

$$\begin{aligned} W_{T,R}(v) = & \quad 1 + \textstyle\sum_{v' \in \mathit{child}_E(v)} W_{T,R}(v') && \text{(by Definition 18)} \\ \geq & \quad 1 + \textstyle\sum_{v' \in \mathit{child}_E(v)} |\{u \mid u \in (\mathit{des}_{T'}(v') \backslash R)\}| && \text{(by (A10))} \\ = & \quad 1 + |\{u' \mid u' \in (\mathit{des}_{T'}(v) \backslash R)\}|. \\ & && \text{(as every child of } v \text{ is an atom vertex)} \end{aligned}$$

Case 2. Suppose that $v \in R$. Then, we can derive the following.

$$\begin{array}{rll} W_{T,R}(v) = & \sum_{v' \in \mathit{child}_E(v)} W_{T,R}(v') & \text{(by Definition 18)} \\ \geq & \sum_{v' \in \mathit{child}_E(v)} |\{u \mid u \in (\mathit{des}_{T'}(v') \backslash R)\}| & \text{(by (A10))} \\ = & |\{u' \mid u' \in (\mathit{des}_{T'}(v) \backslash R\}|. & \\ & & \text{(as every child of } v \text{ is an atom vertex)} \end{array}$$

□

Lemma 16
Let Π be a ground ASP program, X be an answer set for Π, p be an atom in X, $T = \langle V, E, l, \Pi, X \rangle$ be the and-or explanation tree for p with respect to Π and X, v be the root of T, T' be an explanation tree (with vertices V') in T, and R be a set of rule vertices in V. Then,

$$W_{T,R}(v) \geq |\{u \mid u \in (\mathit{RVertices}(T') \backslash R)\}|.$$

Proof of Lemma 16
We want to show that $W_{T,R}(v)$ is equal to at least the number of rule vertices in V' but R. Recall that v is the root of T' and there exists exactly one vertex $v' \in V'$ such that $v' \in \mathit{child}_E(v)$ (due to Condition (iii) in Definition 3). Then, we consider two cases.

Case 1. Assume that $v' \notin R$. Then, we can derive the following.

$$\begin{array}{rll} W_{T,R}(v) = & \mathit{max}\{W_{T,R}(c) \mid c \in \mathit{child}_E(v)\} & \text{(by Definition 18)} \\ \geq & W_{T,R}(v') & (\text{as } v' \in \mathit{child}_E(v)) \\ \geq & 1 + |\{v'' \mid v'' \in (\mathit{des}_{T'}(v') \backslash R)\}| & \text{(by Lemma 15)} \\ = & |\{u \mid u \in (\mathit{RVertices}(T') \backslash R)\}|. & (\text{as } v' \text{ is the only child of } v \text{ in } T') \end{array}$$

Case 2. Assume that $v' \in R$. Then, we can derive the following.

$$\begin{array}{rll} W_{T,R}(v) = & \mathit{max}\{W_{T,R}(c) \mid c \in \mathit{child}_E(v)\} & \text{(by Definition 18)} \\ \geq & W_{T,R}(v') & (\text{as } v' \in \mathit{child}_E(v)) \\ \geq & |\{v'' \mid v'' \in (\mathit{des}_{T'}(v') \backslash R)\}| & \text{(by Lemma 15)} \\ = & |\{u \mid u \in (\mathit{RVertices}(T') \backslash R)\}|. & (\text{as } v' \text{ is the only child of } v \text{ in } T') \end{array}$$

□

Lemma 17
Let Π be a ground ASP program, X be an answer set for Π, p be an atom in X, T be the and-or explanation tree (with vertices V) for p with respect to Π and X, v be the root of T, $T' = \langle V', E', l, \Pi, X \rangle$ be an explanation for p with respect to Π and X, and R be a set of rule vertices in V. Then, $W_{T,R}(v) \geq |\mathit{RVertices}(T') \backslash R|$.

Proof of Lemma 17
We show that $W_{T,R}(v)$ is equal to at least $|\mathit{RVertices}(T')\backslash R|$. By Definition 4, there exists an explanation tree T'' (with vertices V'') in T such that $V' = \{v' \mid v' \text{ is a rule vertex in } V''\}$. That is, $\mathit{RVertices}(T') = \mathit{RVertices}(T'')$. By Lemma 16, we know that $W_{T,R}(v) \geq |\{u \mid u \in (\mathit{RVertices}(T'')\backslash R)\}|$. Thus, we conclude that $W_{T,R}(v) \geq |\mathit{RVertices}(T')\backslash R|$.
□

We can now prove our main result which simply indicates that at each iteration i of the loop in Algorithm 5 the distance $\Delta_D(R_{i-1}, K_i)$ is maximized.

Proposition 7
Let Π be a ground ASP program, X be an answer set for Π, p be an atom in X, and k be a positive integer. Let n be the number of explanations for p with respect to Π and X. Then, at the end of each iteration i ($1 \leq i \leq \mathit{min}\{n, k\}$) of the loop in Algorithm 5, $\Delta_D(R_{i-1}, \mathit{RVertices}(K_i))$ is maximized, i.e., there is no other explanation K' such that $\Delta_D(R_{i-1}, \mathit{RVertices}(K_i)) < \Delta_D(R_{i-1}, \mathit{RVertices}(K'))$.

Proof of Proposition 7
The proof is by contradiction. Assume that there exists an explanation K' such that $\Delta_D(R_{i-1}, \mathit{RVertices}(K_i)) < \Delta_D(R_{i-1}, \mathit{RVertices}(K'))$. That is, $|\mathit{RVertices}(K')\backslash R_{i-1}| > |\mathit{RVertices}(K_i)\backslash R_{i-1}|$. Let v_r be the root of T. Then, by Lemma 17, $W_{T,R_{i-1}}(v_r) \geq |\mathit{RVertices}(K')\backslash R_{i-1}|$. Also, by Lemma 14, we know that $W_{T,R_{i-1}}(v_r) = |\mathit{RVertices}(K_i)\backslash R_{i-1}|$. Therefore, we obtain that $|\mathit{RVertices}(K_i)\backslash R_{i-1}| \geq |\mathit{RVertices}(K')\backslash R_{i-1}|$. But, that contradicts the assumption $|\mathit{RVertices}(K')\backslash R_{i-1}| > |\mathit{RVertices}(K_i)\backslash R_{i-1}|$. Thus, there is no K' such that $\Delta_D(R_{i-1}, \mathit{RVertices}(K_i)) < \Delta_D(R_{i-1}, \mathit{RVertices}(K'))$, i.e., $\Delta_D(R_{i-1}, \mathit{RVertices}(K_i))$ is maximized.
□

Now, we provide the proof of the corollary that shows how to compute longest explanations.

Corollary 1
Let Π be a ground ASP program, X be an answer set for Π, p be an atom in X, and $k = 1$. Then, Algorithm 5 computes a longest explanation for p with respect to Π and X.

Proof of Corollary 1
Since $k = 1$, the loop in Algorithm 5 iterates once. At that iteration, by Proposition 6, we know that an explanation K_1 for p with respect to Π and X is computed. By Proposition 7, we also know that $\Delta_D(R_0, \mathit{RVertices}(K_1))$ is maximized. That is, there exists no explanation K' for p with respect to Π and X such that $|\mathit{RVertices}(K')\backslash R_0| > |\mathit{RVertices}(K_1)\backslash R_0|$. Note that R_0 is an empty set. Therefore, there exists no explanation K' for p with respect to Π and X such that $|\mathit{RVertices}(K')| > |\mathit{RVertices}(K_1)|$. As every vertex of an explanation is a rule vertex, we conclude that K_1 is a longest explanation for p with respect to Π and X.
□

Next, we indicate the proof of the corollary that shows Algorithm 5 computes *min*$\{n, k\}$ different explanations such that for every i ($1 \leq i \leq$ *min*$\{n, k\}$) the i^{th} explanation is the most distant explanation from the previously computed $i - 1$ explanations.

Corollary 2
Let Π be a ground ASP program, X be an answer set for Π, p be an atom in X, and k be a positive integer. Let n be the number of explanations for p with respect to Π and X. Then, Algorithm 5 computes *min*$\{n, k\}$ different explanations $K_1, \ldots, K_{\textit{min}\{n,k\}}$ for p with respect to Π and X such that for every j ($2 \leq j \leq$ *min*$\{n, k\}$) $\Delta_D(\bigcup_{z=1}^{j-1} \textit{RVertices}(K_z), K_j)$ is maximized.

Proof of Corollary 2
By Proposition 6, we know that $K_1, \ldots, K_{\textit{min}\{n,k\}}$ are *min*$\{n, k\}$ different explanations for p with respect to Π and X. Also, by Proposition 7, for every i ($2 \leq i \leq$ *min*$\{n, k\}$), we obtain that $\Delta_D(R_{i-1}, K_i)$ is maximized. Due to Lines 1 and 10 of Algorithm 5, $R_{i-1} = \bigcup_{j=0}^{i-1} \textit{RVertices}(K_j)$. Since R_0 is an empty set, we conclude that $\Delta_D(\bigcup_{j=1}^{i-1} \textit{RVertices}(K_j), K_i)$ is maximized.
□

A.2.4 Proof of Proposition 8 – Complexity of Algorithm 5

We prove Proposition 8 which shows that the time complexity of Algorithm 5 is exponential in the size of the given answer set.

Proposition 8
Given a ground ASP program Π, an answer set X for Π, an atom p in X and a positive integer k, the time complexity of Algorithm 5 is $O(k \times |\Pi|^{|X|+1} \times |\mathcal{B}_\Pi|)$.

Proof of Proposition 8
Algorithm 5 calls Algorithm 2 at Line 2. In the proof of Proposition 4, it is shown that the worst case time complexity of Algorithm 2 is $O(|\Pi|^{|X|} \times \mathcal{B}_\Pi)$. Moreover, Algorithm 5 has a loop between Lines 4–10, which iterates at most k times. In every iteration of the loop, Algorithm 6 and Algorithm 4 are called at Lines 5 and 7. Algorithm 6 simply traverses the and-or explanation tree T recursively (cf. Lines 2, 3, 8 and 9 in Algorithm 6). The height of T is $O(|X|)$ and the branching factor of a vertex in T is $O(|\Pi|)$. Also, at Line 8 in Algorithm 6, we check whether a rule vertex in V is in R. As R is a subset of the rule vertices in V, this check can be done in $O(|\Pi| \times |\mathcal{B}_\Pi|)$ time. Thus, the time complexity of Algorithm 6, in the worst case, is $O(|\Pi|^{|X|} \times |\Pi| \times |\mathcal{B}_\Pi|)$. Similar to Algorithm 6, Algorithm 4 just traverses a portion of T (cf. Lines 3, 5, 9 and 10 in Algorithm 4). Thus, the time complexity of every iteration of the loop in Algorithm 5 is $O(|\Pi|^{|X|+1} \times |\mathcal{B}_\Pi|)$. As the loop iterates at most k times, the time complexity of Algorithm 5, in the worst case, is $O(k \times |\Pi|^{|X|+1} \times |\mathcal{B}_\Pi|)$.
□

A.3 Relations between Explanations and Justifications

In Section 10 of the paper, we have related explanations to justifications, resulting in Propositions 10 and 11. In the following, we show the proofs of these results.

A.3.1 Proof of Proposition 10 – Soundness of Algorithm 7

In the proof of Proposition 10, the idea is to show that Algorithm 7 returns at Line 17 an explanation tree T' in the and-or explanation tree for p with respect to Π and X. That is, T' satisfies Conditions (i)–(iv) in Definition 3. Before providing the proof of Proposition 10, we consider some useful lemmas and corollaries.

Lemma 18
Let Π be a ground normal ASP program, X be an answer set for Π, p be an atom in X, U be an assumption in $Assumptions(\Pi, X)$, $G = (V, E)$ be an offline justification of p^+ with respect to X and U, $T = \langle V', E', l, \Pi, X\rangle$ be an output of Algorithm 7 and $P = \langle v_1, \ldots, v_n\rangle$ be a path in T. Take any three consecutive elements v_i, v_{i+1} and v_{i+2} in P such that v_i and v_{i+2} are atom vertices and v_{i+1} is a rule vertex. Then, $(l(v_i)^+, l(v_{i+2})^+, +) \in E$ holds.

Proof of Lemma 18
As v_i is an atom vertex, its out-going edges are formed at Line 15 of Algorithm 7. Then, due to Lines 13 and 14, v_{i+1} is a rule vertex such that $B(l(v_{i+1})) = support(l(v_i)^+, G)$. As v_{i+1} is a rule vertex, its out-going edges are formed at Line 10. Then, due to Lines 8 and 9, v_{i+2} is an atom vertex where $l(v_{i+2}) \in B^+(l(v_{i+1}))$. Since $B(l(v_{i+1})) = support(l(v_i)^+, G)$ and $l(v_{i+2}) \in B^+(l(v_{i+1}))$, by Definition 10, $(l(v_i)^+, l(v_{i+2})^+, +) \in E$ holds.
□

Corollary 3
Let Π be a ground normal ASP program, X be an answer set for Π, p be an atom in X, U be an assumption in $Assumptions(\Pi, X)$, $G = (N, E)$ be an offline justification of p^+ with respect to X and U, $T = \langle V', E', l, \Pi, X\rangle$ be an output of Algorithm 7 and $P = \langle v_1, \ldots, v_n\rangle$ be a path in T where v_1 and v_n are atom vertices. Then, $l(v_n)^+$ is reachable from $l(v_1)^+$ by a positive path in G.

Proof of Corollary 3
Note that an edge in E' is either defined from an atom vertex to a rule vertex or vice versa due to Lines 10 and 15 of Algorithm 7. By this observation, in P, as v_1 is an atom vertex, v_i is an atom vertex (resp., rule vertex) if i is an odd number (resp., a even number). Then, by Lemma 18, for $1 \leq i \leq n-2$ and $i \mod 2 \neq 0$ (i.e., i is an odd number), $(l(v_i)^+, l(v_{i+2})^+, +) \in E$. Thus, there exists a positive path $\langle l(v_1)^+, l(v_3)^+, l(v_5)^+, \ldots, l(v^+_{n-2}), l(v_n)^+\rangle$ in G. That is, $l(v_n)^+$ is reachable from $l(v_1)^+$ by a positive path in G.
□

Lemma 19
Let Π be a ground normal ASP program, X be an answer set for Π, p be an atom in X, U be an assumption in $Assumptions(\Pi, X)$, $G = (N, E)$ be an offline justification of p^+ with respect to X and U, $T = \langle V', E', l, \Pi, X\rangle$ be an output of Algorithm 7 and v be an atom vertex in V' such that v is the first element added into the queue Q in Algorithm 7. Consider a sequence $S = \langle v_1 = v, v_2, \ldots, v_n\rangle$ of n elements where $v_i \in V'$ for $1 \leq i \leq n$

such that v_{j+1} is added into Q right after v_j for $1 \leq j < n$. Then, every vertex $v_i \in V'$ $(1 < i \leq n)$ is reachable from v by a path in T.

Proof of Lemma 19
The proof is by induction on the length of S.

Base case: Assume that S has two elements, i.e., $S = \langle v_1 = v, v_2 \rangle$. Since v is the first vertex added into Q by definition, it is the first vertex dequeued from Q at Line 5. Then, since v is an atom vertex by definition, the second vertex is added into Q at Line 16. By definition of S, v_2 is the second vertex added into Q. Thus, by Line 15, $(v, v_2) \in E'$, i.e., v_2 is reachable from v by a path in T.

Induction step: As an induction hypothesis, assume that for a sequence $S' = \langle v_1 = v, v_2, \ldots, v_k \rangle$ of k elements, where $v_i \in V'$ for $1 \leq i \leq k$ such that v_{j+1} is added into Q right after v_j for $1 \leq j < k$, every vertex $v_l \in V$ $(1 < l \leq k)$ is reachable from v by a path in T. Let $S'' = \langle v_1 = v, v_2, \ldots, v_{k+1} \rangle$ be a sequence of $k+1$ elements, where $v_i \in V'$ for $1 \leq i \leq k+1$ such that v_{j+1} is added into Q right after v_j for $1 \leq j < k+1$. We show that v reaches every vertex in S'' by using edges in E'. Consider the subsequence S''_{sub} of S'' that consists of the first k elements of S'', i.e., a prefix of S'' with length k. By the induction hypothesis, every vertex $v_i \in S''_{sub}$ is reachable from v by a path in T. Now, we show that v_{k+1} is reachable from v by a path in T. We consider two cases.

Case 1. Assume that v_{k+1} is an atom vertex. Then, v_{k+1} is added into V' at Line 11. So, by Line 10, there exists a vertex v' such that $(v', v_{k+1}) \in E'$. But, due to Line 5, v' must be added into Q prior to v_{k+1}. That is, $v' \in S''_{sub}$. So, by the induction hypothesis, v reaches v' by a path P in T. Thus, by P and (v', v_{k+1}), v_{k+1} is reachable from v by a path in T.

Case 2. Assume that v_{k+1} is a rule vertex. Then, v_{k+1} is added into V' at Line 16. So, by Line 15, there exists a vertex v' such that $(v', v_{k+1}) \in E'$. But, due to Line 5, v' must be added into Q prior to v_{k+1}. That is, $v' \in S''_{sub}$. So, by the induction hypothesis, v reaches v' by a path P in T. Thus, by P and (v', v_{k+1}), v_{k+1} is reachable from v by a path in T.

□

We now prove Proposition 10 that shows the soundness of Algorithm 7.

Proposition 10
Given a ground normal ASP program Π, an answer set X for Π, an atom p in X, an assumption U in $Assumption(\Pi, X)$, and an offline justification $G = (V, E)$ of p^+ with respect to X and U, Algorithm 7 returns an explanation tree $\langle V', E', l, \Pi, X \rangle$ in the and-or explanation tree for p with respect to Π and X.

Proof of Proposition 10
We show what Algorithm 7 returns at Line 17, $T' = \langle V', E', l, \Pi, X \rangle$, is an explanation tree in the and-or explanation tree $T = \langle V_T, E_T, l, \Pi, X \rangle$ for p with respect to Π and X. That is, T' satisfies Conditions (i)–(iv) in Definition 3. In the following, we study each condition separately.

(ii) To show that the root of $\langle V', E' \rangle$ is a vertex whose label is p, we need to show three cases hold; (1) there exists a vertex v in V' with label p, (2) every vertex $v' \in V'$ is reachable from v by a path in T'. (3) v has no in-going edge. In the following, we show that each case holds.

Case 1. Observe that a vertex is in V' if and only if it is added into the queue Q. Then, due to Lines 2 and 3, there exists a vertex $v \in V'$ such that $l(v) = p$.

Case 2. The first element added into Q is an atom vertex v with $l(v) = p$, due to Lines 2 and 3. Note that a vertex is in V' if and only if it is added into Q. Then, as v is added into Q, by the observation, $v \in V'$. Now, pick a vertex $v' \in V'$. By the same observation, v' is also added into Q. Since v is the first element queued in Q, v' is queued in V' later on. Thus, by Lemma 19, v' is reachable from v by a path in T'.

Case 3. Assume otherwise. That is, $(v', v) \in E'$ for some vertex $v' \in V'$. The edges in E' are constructed at Lines 10 and 15. Then, as v is an atom vertex, v' is a rule vertex. Observe that a vertex is in V' if and only if it is added into Q. Then, since v' is a rule vertex in V', it is added into Q at Line 16. So, due to Line 15, there exists an atom vertex $v'' \in V'$ such that $(v'', v') \in E'$. Thus, as $(v'', v'), (v', v) \in E'$, there exists a path $\langle v'', v', v \rangle$ in T'. Then, by Corollary 3, $l(v)^+$ is reachable from $l(v'')^+$ by a positive path P in G. Moreover, as shown in Case 2 above, v'' is reachable from v by a path $\langle v_1 = v, v_2, \ldots, v_n = v'' \rangle$ in T'. So, by Corollary 3, $l(v'')^+$ is reachable from $l(v)^+$ by a positive path P' in G. Then, by P and P', a positive cycle exists in G. As every offline justification is a safe offline e-graph due to Definition 17, we reach a contradiction.

(i) By Condition (ii) in Definition 3, we know that the root of $\langle V', E' \rangle$ is a vertex v with $l(v) = p$. Then, if we show that for every vertex $v' \in V'$, $\mathit{out}_{E'}(v')$ is a subset of E_T, then we can conclude that $\langle V', E' \rangle$ is a subtree of $\langle V_T, E_T \rangle$. Now, pick a vertex $v' \in V'$. We consider two cases:

Case 1. Assume that v' is an atom vertex. Take an out-going edge (v', v'') of v' in E'. To show that $(v', v'') \in E_T$, we need to show that $l(v'') \in \Pi_{X, \mathit{anc}_{T'}(v'')}(l(v'))$, due to Condition (ii) in Definition 2. For that, we should show that $H(l(v'')) = l(v')$, $B^+(l(v'')) \subseteq X \backslash \mathit{anc}_{T'}(v'')$, $B^-(l(v'')) \cap X = \emptyset$ and $X \models B_{card}(l(v''))$, due to (8) in Section 4. As Π is a ground normal ASP program, it does not contain cardinality expressions in its body. Thus, $X \models B_{card}(l(v''))$ trivially. In Algorithm 7, outgoing edges of atom vertices are formed at Line 15. Then, due to Lines 13 and 14, $H(l(v'')) = l(v')$ and $B(l(v'')) = \mathit{support}(l(v')^+, G)$. Since G is an offline justification, G is an offline e-graph by Definition 17. Then, by Definition 13, G is an (X, U)-based e-graph. So, by Definition 12, $\mathit{support}(l(v')^+, G)$ is an LCE of $l(v')^+$ with respect to (X, U), so does $B(l(v''))$. According to Definition 11, as $B(l(v'')) \neq \{assume\}$, $B^+(l(v'')) \subseteq X$ and $B^-(l(v'')) \cap X = \emptyset$. To complete this part, it remains to show that $B^+(l(v'')) \subseteq X \backslash \mathit{anc}_{T'}(v'')$. Since $B^+(l(v'')) \subseteq X$, it is enough to show that $B^+(l(v'')) \cap \mathit{anc}_{T'}(v'') = \emptyset$. For that, assume $B^+(l(v'')) \cap \mathit{anc}_{T'}(v'') \neq \emptyset$. Let $a \in B^+(l(v'')) \cap \mathit{anc}_{T'}(v'')$. Since $a \in B^+(l(v''))$ and $B(l(v'')) = \mathit{support}(l(v')^+, G)$, $(l(v')^+, a^+, +) \in E$ holds. As $a \in \mathit{anc}_{T'}(v'')$, there exists a vertex $u \in V'$ where $l(u) = a$ such that a path $P = \langle v_1 = u, \ldots, v_n = v' \rangle$ in T' exists. Then, due to Corollary 3, $l(v')^+$ is reachable from a^+ by a positive path in G. But, as $(l(v')^+, a^+, +) \in E$, $(a^+, a^+) \in E^{*,+}$

holds, i.e., there is a positive cycle in the offline justification, which is a contradiction, due to Definition 17.

Case 2. Assume that v' is a rule vertex. Take an out-going edge (v', v'') of v' in E'. To show that $(v', v'') \in E_T$, we need to show that $l(v'') \in B^+(l(v'))$, due to Condition (iii) in Definition 2. Out-going edges of v' are formed at Line 10 which is reached only by satisfying the condition at Line 8. Then, due to Line 9, $l(v'') \in B^+(l(v'))$.

(iii) Observe that an element is added into Q once. Thus, we dequeue each atom vertex only once. Then, due to Lines 12–16, every atom vertex $v' \in V'$ has exactly one out-going edge, i.e., $deg_{E'}(v') = 1$.

(iv) Take a rule vertex $v' \in V'$. Its edges are formed at Line 10. Then, due to the condition at Line 8 and the statement at Line 9, for every $a \in B^+(l(v'))$, there exists a vertex v'' such that $l(v'') = a$. Then, the condition holds.

□

Also, we can show that the and-or explanation tree exists for every atom in an answer set.

Proposition 1
Let Π be a ground ASP program and X be an answer set for Π. For every p be in X, the and-or explanation tree for p with respect to Π and X is not empty.

Proof of Proposition 1
By Proposition 9, we know that there exists an offline justification of p^+ with respect to X and $X^- \backslash WF_\Pi^-$. Then, by Proposition 10, there is an explanation tree in the and-or explanation tree for p with respect to Π and X. That is, the and-or explanation tree for p with respect to Π and X is not empty.

□

A.3.2 Proof of Proposition 11 – Soundness of Algorithm 8

Before proving Proposition 11, we provide the necessary lemmas.

Lemma 20
Let Π be a ground normal ASP program, X be an answer set for Π, p be an atom in X, $T = \langle V', E', l, \Pi, X \rangle$ be an explanation tree in the and-or explanation tree for p with respect to Π and X such that for every $v, v' \in V'$, $l(v) = l(v')$ if and only if $v = v'$, (V, E) be the output of Algorithm 8 called with inputs Π, X, p and T, and $\langle v_1, v_2, v_3 \rangle$ be a path in T such that $l(v_1)^+ \in V$. Then, $(l(v_1)^+, l(v_3)^+, +) \in E$ and $l(v_3)^+ \in V$.

Proof of Lemma 20
All of the nodes, except the one with label $\top$, are added to V at Line 5 of Algorithm 8. As $l(v_1)^+ \in V$, v_1 is extracted from the queue Q at Line 4. Then, since $\langle v_1, v_2, v_3 \rangle$ is a path in T and every atom vertex in V' has exactly one child due to Condition (iii) in Definition 3, v_2 is obtained at Line 6. By the condition at Line 9, every child of v_2 is considered in the loop. Accordingly, at Line 10, the edge $(l(v_1)^+, l(v_3)^+, +)$ is added into E. Also, v_3 is

added into Q at Line 11. Then, due to the condition at Line 3 and the statements at Lines 4 and 5, $l(v_3)^+ \in V$.

□

Corollary 4
Let Π be a ground normal ASP program, X be an answer set for Π, p be an atom in X, $T = \langle V', E', l, \Pi, X\rangle$ be an explanation tree in the and-or explanation tree for p with respect to Π and X such that for every $v, v' \in V'$, $l(v) = l(v')$ if and only if $v = v'$, (V, E) be the output of Algorithm 8 called with inputs Π, X, p and T, and $\langle v_1, v_2, \ldots, v_n\rangle$ be a path in T such that $l(v_1)^+ \in V$. Then, $\langle l(v_1)^+, l(v_3)^+, l(v_5)^+, \ldots, l(v_n)^+\rangle$ is a path in (V, E).

Proof of Corollary 4
Since $\langle v_1, v_2, v_3\rangle$ is a path in T and $l(v_1)^+ \in V$, by Lemma 20, $(l(v_1)^+, l(v_3)^+, +) \in E$ and $l(v_3)^+ \in V$. Similarly, $(l(v_3)^+, l(v_5)^+, +) \in E$ and $l(v_5)^+ \in V$. Then, this incremental application of Lemma 20 leads that $\langle l(v_1)^+, l(v_3)^+, l(v_5)^+, \ldots, l(v_n)^+\rangle$ is a path in (V, E).

□

We now prove Proposition 11 which shows that Algorithm 8 creates an offline justification of the given atom in the reduct of the given ASP program with respect to the given answer set, provided that labels of the vertices of the given explanation tree are unique labels, i.e., no two different vertices labels the same entity.

Proposition 11
Given a ground normal ASP program Π , an answer set X for Π, an atom p in X, and an explanation tree $\langle V', E', l, \Pi, X\rangle$ in the and-or explanation tree for p with respect to Π and X such that for every $v, v' \in V'$, $l(v) = l(v')$ if and only if $v = v'$, Algorithm 8 returns an offline justification of p^+ in Π^X with respect to X and $\emptyset$.

Proof of Proposition 11
To show that the output (V, E) of Algorithm 8 at Line 13 is an offline justification of p^+ in Π^X with respect to X and $\emptyset$, one needs to show that the following conditions hold.

(1) $\emptyset \in Assumptions(\Pi^X, X)$;
(2) (V, E) is an e-graph for Π^X;
(3) (V, E) is a $(X, \emptyset)$-based e-graph of p^+;
(4) (V, E) is an offline e-graph of p^+ with respect to X and $\emptyset$.

Condition (1) We show that $\emptyset \in Assumptions(\Pi^X, X)$. As Π^X is a positive program, i.e., it does not contain any negative atoms, by Definition 14, $\mathcal{TA}_{\Pi^X}(X) = \emptyset$. Also, due to Definition 15, $NR(\Pi^X, X) = \Pi^X$. Then, by Definition 16, $\emptyset \in Assumptions(\Pi^X, X)$.

Condition (2) We show that (V, E) is an e-graph for Π^X. For that, (V, E) should satisfy Conditions $(i) - (iv)$ in Definition 9.

(i) Consider two cases.

Case 1. Take a node $b^+ \in V \backslash \{\top\}$. It is added to V at Line 5. Then, there exists an atom vertex $v \in V'$ such that $l(v) = b$. By Condition (iii) in Definition 3, v has exactly one child v' in $\langle V', E' \rangle$, which is a rule vertex. Then, depending on whether $l(v')$ is a fact or not, an out-going edge of b^+ in E is formed at Line 8 or 10. So, b^+ is not a sink.
Case 2. Let l be $\top$. Then, it is added to V at Line 12. The edges in E are formed at Lines 8 and 10. Accordingly, l has no out-going edge, i.e., it is a sink.
Therefore, (V, E) is an e-graph for Π^X.

(ii) Due to Lines 8 and 10, clearly, for every $b \in V$ there is no edges in the form of $(b, assume, -)$ and $(b, \bot, -)$ in E.

(iii) Similar to (ii), for every $b \in V$ there is no edges in the form of $(b, assume, +)$ and $(b, \top, +)$ in E.

(iv) Let $b^+ \in V$ such that $(b^+, \top, +) \in E$. As out-going edges are created at Lines 8 and 10, $(b, \top, +)$ must be formed at Line 8. Then, there exists an atom vertex $v \in V'$ such that $l(v) = b$ and the label of the child v' of v is a fact (Line 7). Then, since the condition at Line 9 is not satisfied, it is not possible that b^+ has another out-going edge.

Condition (3) We show that (V, E) is an $(X, \emptyset)$-based e-graph of p^+. By Condition (1) above, we know that (V, E) is an e-graph. To show that (V, E) is an $(X, \emptyset)$-based e-graph of p^+, we need to show that Conditions (i) and (ii) in Definition 12 hold.

(i) Take a node $c^+ \in V$. It is added to V at Line 5. Then, there exists an atom vertex $v' \in V'$ such that $l(v') = c$. By Condition (ii) in Definition 3, we know that the root of $\langle V', E' \rangle$ is a vertex v where $l(v) = p$. So, v' is reachable from v by a path $\langle v_1 = v, v_2, v_3, \ldots, v_n = v' \rangle$ in $\langle V', E' \rangle$. Also, as v is added into Q at Line 2, we know that $l(v)^+ \in V$ by Line 5. Then, by Corollary 4, $\langle l(v)^+, l(v_3)^+, \ldots, l(v')^+ \rangle$ is a path in (V, E). That is, c^+ is reachable from p^+.

(ii) Let $G = (V, E)$. Take a node $c^+ \in V \backslash \{\top\}$. Then, $support(c^+, G) = \{a \mid (c^+, a^+, +) \in E\}$ or $support(c^+, G) = \{\top\}$.
Assume that $support(c^+, G) = \{a \mid (c^+, a^+, +) \in E\}$. As $c^+ \in V \backslash \{\top\}$, there exists an atom vertex $v \in V'$ such that $l(v) = c$. By Condition (iii) in Definition 3, there exists exactly one child v' of v in $\langle V', E' \rangle$. Then, due to the condition at Line 9, for each $a \in support(c^+, G)$, $(v', v'') \in E'$ where $v'' \in V'$ with $l(v'') = a$. Thus, $B^+(l(v')) = support(c^+, G)$. By Condition (ii) in Definition 2, $H(l(v')) = c$ and $B^+(l(v')) \subseteq X$. Hence, $support(c^+G)$ is an LCE of $(X, \emptyset)$.

Condition (4) We show that (V, E) is an offline e-graph of p^+ with respect to X and $\emptyset$. By Condition (3) above, we know that (V, E) is an $(X, \emptyset)$-based e-graph of p^+. Then, to show that (V, E) is an offline e-graph of p^+ with respect to X and $\emptyset$, we need to show that (V, E) satisfies Conditions (i) and (ii) in Definition 13. As edges in E are formed at Lines 8 and 10, for every $b \in V$ there are no edges in the form of $(b, assume, +)$ and $(b, assume, -)$ in E. Then, conditions are trivially satisfied.

□