Online appendix for the paper

*Complexity and Compilation of GZ-Aggregates in Answer Set Programming*

published in Theory and Practice of Logic Programming

MARIO ALVIANO and NICOLA LEONE

Department of Mathematics and Computer Science, University of Calabria, Italy

submitted 30 June 2015; revised 04 July 2015; accepted 14 July 2015

Appendix A Proofs of Section 3

Lemma 1
Let Π be in ASP(M). The least fixpoint of $T_Π(I)$ exists and is polytime computable. Let $I$ be the least fixpoint of $T_Π$, and $J$ be the least fixpoint of $T_{G(Π,I)}$. If $I \neq J$ then Π is G-incoherent, otherwise $GSM(Π) = \{I\}$.

Proof
We first show that the least fixpoint of $T_Π$ is polytime computable. Let Π be a program in ASP(M), and $I$ be an interpretation. Computing $T_Π(I)$ requires to iterate over every rule $r$ of Π and check whether $I \models B(r)$. Checking $I \models B(r)$ can be done in polynomial-time if aggregates are polynomial-time computable functions, as it is assumed in this section. Hence, a single application of $T_Π$ is polynomial-time computable. The least fixpoint of $T_Π$ is computed, by definition, starting from $∅$ and repeatedly applying $T_Π$. Define $I_0 = ∅$, $I_{i+1} = T_Π(I_i)$ (for $i ≥ 0$). For each $i ≥ 0$, either $I_{i+1} \setminus I_i \neq ∅$ or $I_i$ is the least fixpoint of $T_Π$. Since atoms in $I_{i+1} \setminus I_i$ are among those in $At(Π)$, we have that $I_{|At(Π)|} = I_{|At(Π)|+1}$.

We now show the second part of the lemma. $I \models Π$ by construction. Note that $G(Π,I)$ is a plain Datalog program. It is unique minimal model is the least fixpoint of $T_{G(Π,I)}$, i.e., interpretation $J$. Hence, $I \in GSM(Π)$ if and only if $I = J$. To complete the proof is enough to show that no other interpretation is a G-stable model of Π. Let $K$ be an interpretation such that $K \neq I$ and $K \models Π$. Therefore, $K \supset I$ because $I$ is the least fixpoint of $T_Π$. To prove that $K \notin GSM(Π)$ note that $I \models G(Π,K)$. □

Theorem 1
G-coherence testing is in P for ASP(M).
Proof
Let $I$ be the least fixpoint of $T_\Pi$. $I$ is computable in polynomial-time because of Lemma 1. Actually, $I$ is the only candidate to be a G-stable model of $\Pi$ because of Lemma 1. To check whether $I \in GSM(\Pi)$, build $G(\Pi, I)$ and compute the least fixpoint of $T_{G(\Pi, I)}$, again in polynomial-time because of Lemma 1. If the two least fixpoints are equal then $\Pi$ is G-coherent, otherwise it is G-incoherent.

Theorem 2
G-coherence testing is in NP for programs in ASP($\neg$, M, C, N).

Proof
Let $\Pi$ be in ASP($\neg$, M, C, N), and $I$ be an interpretation. We provide a polynomial-time procedure for checking that $I$ is a G-stable model of $\Pi$. The procedure first checks that $I \models \Pi$ in polynomial-time. If it is the case, the procedure builds the reduct $G(\Pi, I)$, again in polynomial-time. Program $G(\Pi, I)$ is in ASP($\neg$) and therefore Lemma 1 can be applied to obtain the unique minimal model of $G(\Pi, I)$, say $J$, in polynomial-time. If $I = J$ then the procedure accepts $I$ as a G-stable model, otherwise it rejects $I$.

Lemma 2
Let $\Pi$ be in ASP($\neg$, $\lor$). Then, $GSM(\Pi) \equiv_{At(\Pi)} GSM(C(\Pi)) \equiv_{At(\Pi)} GSM(N(\Pi))$.

Proof
Let $I$ be an interpretation. $I \models \Pi$ if and only if $I \models C(\Pi)$. In particular, if $\neg p$ is replaced by an aggregate $A$ in a rule $r$, we have $I \nmid \neg p$ if and only if $I \models A$. Note that $I \nmid \neg p$ implies that $r$ is removed in the reducts $G(\Pi, I), G(C(\Pi), I)$, while $I \models \neg p$ implies that both $\neg p$ and $A$ are replaced by the empty set in the rules obtained from $r$ in the reducts. We therefore conclude that $G(\Pi, I) = G(\Pi, C(I))$, from which we obtain $GSM(\Pi) \equiv_{At(\Pi)} GSM(C(\Pi))$.

The proof of $GSM(\Pi) \equiv_{At(\Pi)} GSM(N(\Pi))$ is similar. We have just to additionally note that $\bot \notin I$ holds for every $I \in GSM(\Pi) \cup GSM(N(\Pi))$.

Theorem 3
G-coherence testing is $\Sigma_2^P$-hard for both ASP($\lor$, C) and ASP($\lor$, N). It is NP-hard for both ASP(C) and ASP(N).

Proof
G-coherence testing is $\Sigma_2^P$-hard for ASP($\neg$, $\lor$), and it is NP-hard for ASP($\neg$) (Eiter and Gottlob 1995). G-coherence of $\Pi$ can be reduced to G-coherence testing of $C(\Pi)$ or of $N(\Pi)$ because of Lemma 2. Since $C(\Pi)$ and $N(\Pi)$ can be computed in polynomial-time, do not introduce disjunction, eliminate negation, and only have convex and non-convex aggregates, respectively, the proof is complete.

Theorem 4
G-coherence testing is P-hard for ASP(M).
Proof

G-cautious reasoning over Datalog programs is P-hard (Eiter and Gottlob 1995). We reduce this problem to G-coherence testing of disjunction- and negation-free programs with monotone aggregates. Let II be in ASP(¬), and p be a propositional atom. Program II′ = II ∪ {p ← A}, where dom(A) = {p} and A(I) = |{p} ∩ I| ≥ 0, can be built using only logarithmic space. Since II is a Datalog program, it has a unique G-stable model, say I. If p ∈ I then p belongs to the least fixpoint of TII because of Lemma 1, and therefore it belongs to the least fixpoint of TII′ too because of monotonicity. On the other hand, if p /∈ I then any model J of II′ is such that J ⊃ I because of rule p ← A (note that A is always true). We conclude that G(II′, J) = G(II, J) ∪ {p ← p}, and therefore the least fixpoint of TG(II′, J), which is equal to the least fixpoint of TG(II, J), is a subset of I. We conclude that J is not a G-stable model of II′ and hence II′ is G-incoherent.

Lemma 3

Let II be in ASP(¬, ∨). The following relation holds: GSM(II) ≡_{At(II)} GSM(M(II)).

Proof

Without loss of generality, let us assume that all atoms in At(II) occur negated in II at least once. Let I be a G-stable model of II. Define IF = I ∪ {pF | p /∈ I}. We have IF |= M(II). Concerning G(M(II), IF) note that for each p ∈ At(II) rule p ∨ pF ← A is either replaced by

p ∨ pF ←

in case p /∈ I, or by

p ∨ pF ← p

if p ∈ I. In the first case, the rule guarantees that every model J of G(M(II), IF) such that J ⊂ I satisfies pF ∈ J. Hence, rules of G(M(II), IF) containing pF can be simplified by removing pF, which essentially results into G(II, I) (plus rules obtained from p ∨ pF ← A). In the second case, the rule is trivially satisfied by all interpretations, and therefore it can be removed from G(M(II), IF). Since I is a minimal model of G(II, IF), we have that IF is a minimal model of G(M(II), IF), i.e., IF ∈ GSM(M(II)).

For the other direction, let I be a G-stable model of M(II). We shall show that I ∩ At(II) is a G-stable model of II. First of all, note that I |= A for any aggregate A occurring in M(II), and therefore I ∩ {p, pF} ≠ ∅ because of rule p ∨ pF ← A, for all p ∈ At(II). Moreover, since I is a minimal model of G(M(II), I) by assumption, and pF does not occur in any other rule heads, we have |I ∩ {p, pF}| = 1. We can therefore argument as in the previous direction and conclude that I ∩ At(II) is a minimal model of G(II, I ∩ At(II)), i.e., I ∩ At(II) ∈ GSM(II).

As a final observation, note that also |GSM(II) |= |GSM(M(II))| holds because in any G-stable model of M(II) truth values for atoms of the form pF are implied by truth values of atoms of the form p.

□
**Theorem 5**

G-coherence testing is $\Sigma_2^P$-hard for ASP($\lor$, $M$).

**Proof**

G-coherence testing is $\Sigma_2^P$-hard for a program $\Pi$ in ASP($\neg$, $\lor$) (Eiter and Gottlob 1995). G-coherence of $\Pi$ can be reduced to G-coherence testing of $M(\Pi)$ because of Lemma 3. Since $M(\Pi)$ can be computed in polynomial-time, eliminates negation, and only has monotone aggregates, the proof is complete. □

**Theorem 6**

G-cautious reasoning is in $P$ for ASP($M$).

**Proof**

We provide a procedure for checking whether a given propositional atom $p$ is a G-cautious consequence of $\Pi$. The procedure first checks G-coherence of $\Pi$ in polynomial-time (Theorem 1). If $\Pi$ is G-incoherent then the procedure rejects. Otherwise, because of Lemma 1, the unique G-stable model of $\Pi$, say $I$, is the least fixpoint of $T_\Pi$. The procedure then computes $I$ in polynomial-time (Lemma 1), and accepts if $p \in I$, otherwise it rejects. □

**Theorem 7**

G-cautious reasoning is in co-$NP$ for programs in ASP($\neg$, $M$, $C$, $N$).

**Proof**

Let $\Pi$ be in ASP($\neg$, $M$, $C$, $N$), and $p$ a propositional atom. We prove that the complementary problem, checking the existence of a G-stable model $I$ of $\Pi$ such that $p \notin I$, is in $NP$. To this aim, let $I$ be an interpretation such that $p \notin I$. The following is a polynomial-time procedure for checking that $I$ is a G-stable model of $\Pi$: The procedure first builds $G(\Pi, I)$, which is disjunction-, negation and aggregate-free. Then, it computes the unique G-stable model, say $J$, of $G(\Pi, I)$, i.e., the least fixpoint of $T_{G(\Pi, I)}$ (Lemma 1), and accepts if $I = J$. □

**Theorem 8**

G-cautious consequence is $\Pi_2^P$-hard for ASP($\lor$, $M$), ASP($\lor$, $C$) and ASP($\lor$, $N$). It is co-$NP$-hard for ASP($C$) and ASP($N$).

**Proof**

G-cautious reasoning is $\Pi_2^P$-hard for ASP($\neg$, $\lor$) already for programs in which negation only occurs in a rule of the form $w \leftarrow \sim w$ (Eiter and Gottlob 1995). Therefore, let us consider a program $\Pi = \Pi' \cup \{w \leftarrow \sim w\}$, where $\Pi'$ is in ASP($\lor$). From Lemmas 2–3, $GSM(\Pi) \equiv_{At(\Pi)} GSM(M(\Pi)) \equiv_{At(\Pi)} GSM(C(\Pi)) \equiv_{At(\Pi)} GSM(N(\Pi))$. Let $p$ be a propositional atom among those in $At(\Pi)$. It holds that $p$ is a G-cautious consequence of $\Pi$ if and only if $p$ is a G-cautious consequence of the other programs. Hence, $\Pi_2^P$-hardness follows.

Similarly, G-cautious reasoning for ASP($\neg$) is co-$NP$-hard already for programs in which negation only occurs in a rule of the form $w \leftarrow \sim w$. Since $C(\Pi)$ and $N(\Pi)$ are disjunction-free if $\Pi$ is disjunction-free, co-$NP$-hardness follows. □
Appendix B  Proofs of Section 4

Theorem 9
Let \( \Pi \) be a program. The following relation holds: \( \text{GSM}(\Pi) \equiv_{\text{At}(\Pi)} \text{FSM}(\text{rew}(\Pi)) \).

Proof
Let \( I \) be a G-stable model of \( \Pi \). We shall show that \( I' = I \cup \{ p' \mid p \in \text{At}(\Pi) \} \) is an F-stable model of \( \text{rew}(\Pi) \). In fact, \( I' \models \text{rew}(\Pi) \) because \( I \models \Pi \). Consider a model \( J \subseteq I \) of the reduct \( F(\text{rew}(\Pi), I) \). We have \( J \cap \text{At}(\Pi) \models G(\Pi, I) \), and therefore \( J \cap \text{At}(\Pi) = I \) holds because \( I \) is a G-stable model of \( \Pi \) by assumption. Because of rules of introduced by item 1 in Definition 4, \( J \cap \text{At}(\Pi) = I \) implies \( J = I \), i.e., \( I \) is an F-stable model of \( \text{rew}(\Pi) \).

Let \( I \) be an F-stable model of \( \text{rew}(\Pi) \). We shall show that \( I \cap \text{At}(\Pi) \) is a G-stable model of \( \Pi \). First of all, note that \( \{ p' \mid p \in \text{At}(\Pi) \} \subseteq I \) because \( I \models \Pi \) and because of rules introduced by item 1 in Definition 4. Therefore, \( I \cap \text{At}(\Pi) \models \Pi \) follows. Consider a model \( J \subseteq I \cap \text{At}(\Pi) \) of the reduct \( G(\Pi, I) \). We have \( J \cup \{ p' \mid p \in \text{At}(\Pi) \} \models F(\text{rew}(\Pi), I) \), and therefore \( J \cup \{ p' \mid p \in \text{At}(\Pi) \} = I \) because \( I \) is an F-stable model of \( \text{rew}(\Pi) \) by assumption. It follows that \( J = I \cap \text{At}(\Pi) \), i.e., \( I \cap \text{At}(\Pi) \) is a G-stable model of \( \Pi \).

Finally, note that also \( |\text{GSM}(\Pi)| = |\text{FSM}(\text{rew}(\Pi))| \) holds because the mappings used above are one-to-one.

Theorem 10
Let \( \Pi \) be a program. The following relation holds: \( \text{GSM}(\Pi) \equiv_{\text{At}(\Pi)} \text{FSM}(\text{str}(\Pi)) \).

Proof
Let \( I \) be a G-stable model of \( \Pi \). We shall show that \( I' = I \cup \{ p' \mid p \in \text{At}(\Pi) \} \cup \{ p'' \mid p \in I \} \) is an F-stable model of \( \text{str}(\Pi) \). In fact, \( I' \models \text{str}(\Pi) \) because \( I \models \Pi \). Consider a model \( J \subseteq I \) of the reduct \( F(\text{str}(\Pi), I) \). We have \( J \cap \text{At}(\Pi) \models G(\Pi, I) \), and therefore \( J \cap \text{At}(\Pi) = I \) holds because \( I \) is a G-stable model of \( \Pi \) by assumption. Because of rules of the group (i)–(ii) in Definition 5, \( J \cap \text{At}(\Pi) = I \) implies \( J = I \), i.e., \( I \) is an F-stable model of \( \text{str}(\Pi) \).

Let \( I \) be an F-stable model of \( \text{str}(\Pi) \). We shall show that \( I \cap \text{At}(\Pi) \) is a G-stable model of \( \Pi \). First of all, note that \( \{ p' \mid p \in \text{At}(\Pi) \} \subseteq I \) because \( I \models \Pi \) and because of rules of the group (i). Moreover, note that \( p \in I \) if and only if \( p'' \in I \) because of rules of the group (iii), for all \( p \in \text{At}(\Pi) \). And also note that for each aggregate \( A'' \) occurring in \( \text{str}(\Pi) \), \( I \models A'' \) if and only if \( I \cap \text{At}(\Pi) \models A \). Therefore, \( I \cap \text{At}(\Pi) \models \Pi \) follows. Consider a model \( J \subseteq I \cap \text{At}(\Pi) \) of the reduct \( G(\Pi, I) \), and define \( J' = J \cup \{ p' \mid p \in \text{At}(\Pi) \} \cup \{ p'' \mid p \in I \} \). We have \( J' \models F(\text{str}(\Pi), I) \), and therefore \( J' = I \) because \( I \) is an F-stable model of \( \text{str}(\Pi) \) by assumption. It follows that \( J = I \cap \text{At}(\Pi) \), i.e., \( I \cap \text{At}(\Pi) \) is a G-stable model of \( \Pi \).

Finally, note that also \( |\text{GSM}(\Pi)| = |\text{FSM}(\text{str}(\Pi))| \) holds because the mappings used above are one-to-one.
**Theorem 11**

Let $\Pi, \Pi'$ be programs such that $\Pi \cap \Pi' = \emptyset$. For $tr \in \{rew, str\}$, the following conditions are satisfied: $tr(\Pi \cup \Pi') = tr(\Pi) \cup tr(\Pi')$, and $tr(\Pi) \cap tr(\Pi') = \emptyset$.

**Proof**

Immediate because the rewritings work on one rule at a time. \hfill $\square$

**Theorem 12**

Let $\Pi$ be a program. The programs $rew(\Pi)$ and $str(\Pi)$ are polynomial-time constructible, and the following relations hold: (i) $\|rew(\Pi)\| \leq 4 \cdot |At(\Pi)| + 2 \cdot \|\Pi\|$; (ii) $\|str(\Pi)\| \leq 10 \cdot |At(\Pi)| + 2 \cdot \|\Pi\|$.

**Proof**

We first prove relation (i). Program $rew(\Pi)$ contains 2 rules for each atom in $At(\Pi)$, each one of size 2, and a rule for each rule of $\Pi$. The number of atoms in these rules is at most twice the number of atoms in the original rules.

We now show relation (ii). Program $rew(\Pi)$ contains 5 rules for each atom in $At(\Pi)$, each one of size 2, and a rule for each rule of $\Pi$. The number of atoms in these rules is at most two times the number of atoms in the original rules. \hfill $\square$

**Theorem 13**

Let $\Pi$ be a program, and $I$ be an interpretation. If $I \models rew(\Pi)$ or $I \models str(\Pi)$ then \{$p' \mid p \in At(\Pi)$\} $\subseteq I$. Moreover, for each $J \subseteq I$ such that $J \models F(str(\Pi), I)$, it holds that \{$p'' \mid p \in I$\} $\subseteq J$.

**Proof of Theorem 13**

If $I$ satisfies rules introduced by item 1 in Definition 4, or equivalently of the group (i) in Definition 5, then \{$p' \mid p \in At(\Pi)$\} $\subseteq I$. Consider a model $J \subseteq I$ of the reduct $F(str(\Pi), I)$. For each $p'' \in I$, $F(str(\Pi), I)$ contains a rule $p'' \leftarrow$ because of rules of the group (ii) in Definition 5. \hfill $\square$

**Theorem 14**

Let $\Pi$ be a program. All aggregates in $str(\Pi)$ are stratified, and if $\Pi$ has no disjunction then both $rew(\Pi)$ and $str(\Pi)$ have no disjunction.

**Proof**

We first provide a more formal definition of stratified aggregate. The dependency graph of $\Pi$ has a node $p$ for each atom $p \in At(\Pi)$, and an arc from $q$ to $p$ if there is a rule $r \in \Pi$ such that $p \in H(r)$ and $q$ occurs in $B(r)$, either as a possibly negated literal or in the domain of an aggregate. $\Pi$ is stratified with respect to aggregates if there is no rule $r \in \Pi$ such that $p \in H(r)$ and $q$ occurring in $B(r)$ belong to the same strongly connected component of $\Pi$.

Let $\Pi$ be a program, and $A$ be an aggregate in $str(\Pi)$. Hence, by construction, $dom(A) \subseteq \{p'' \mid p \in At(\Pi)$\}. Note that all rules whose head contains some atom in $dom(A)$ belong to the group (ii) in Definition 5, and therefore each atom
\( p'' \in \text{dom}(A) \) belongs to a singleton strongly connected component. Stratification of aggregates in \( \text{str}(\Pi) \) is thus proved.

Let \( \Pi \) be a program without disjunction. Program \( \text{rew}(\Pi) \) and \( \text{str}(\Pi) \) contain rules of the groups (i)–(iii), which have no disjunction, and rules obtained from those in \( \Pi \) by replacing aggregates. Hence, neither \( \text{rew}(\Pi) \) nor \( \text{str}(\Pi) \) has disjunction.

References