Splitting a Logic Program Revisited

Jianmin Ji[a], Hai Wan[b+], Ziwei Huo[b], and ZhenFeng Yuan[b]

[a]University of Science and Technology of China, Hefei, China [b]Sun Yat-sen University, Guangzhou, China [+]Email: wanhai@mail.sysu.edu.cn

Problem

Lifschitz and Turner introduced the notion of the splitting set and provided a method to divide a logic program into two parts. They showed that the task of computing the answer sets of the program can be converted into the tasks of computing the answer sets of these parts [1].

However, the empty set and the set of all atoms are the only two splitting sets for many programs, then these programs cannot be divided by the splitting method.

Contributions

we propose a new splitting method that allows the program to be split into two parts by an arbitrary set of atoms, while one of them, the "top" part, may introduce some new atoms.

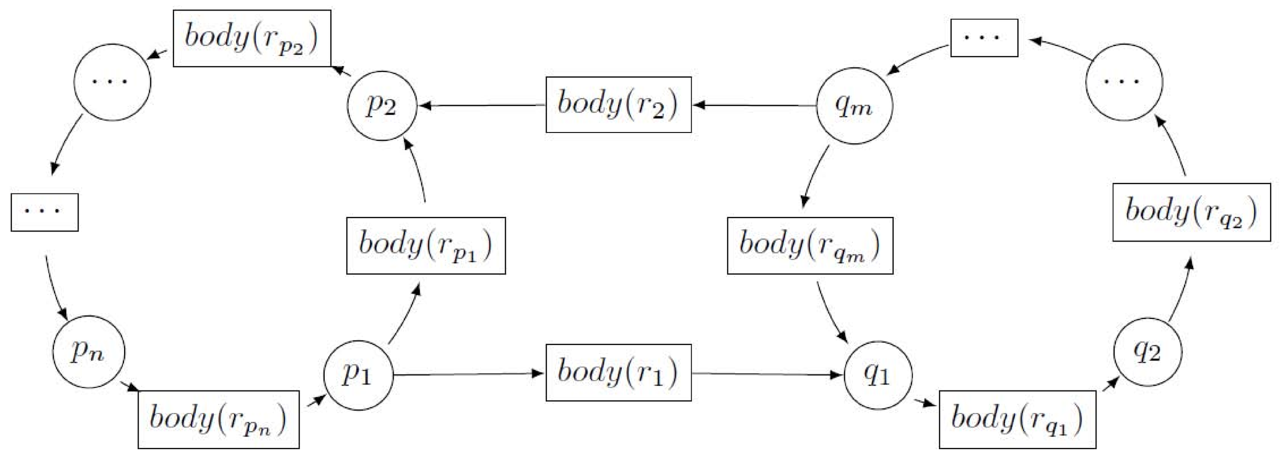
We show that the task of computing the answer sets of the program can be converted into the tasks of computing the answer sets of these parts.

The experiments show that, the splitting result in our method is the same as the result in Lifschitz and Turner's splitting method.

Splitting Sets

just hold on

Splitting with arbitrary set for NLP



By considering the *positive body-head dependency graph* of a program, the method of computing proper loops can be extended and a larger number of loops could be proved to be not proper. We provide an alternative approach.

Intuitively, if we can "remove" the vertex $body(r_1)$ or $body(r_2)$ in left figure, then the number of loops would be greatly reduced. Let the left subgraph be a strongly connected subgraph with n number of atoms and the right subgraph a strongly connected subgraph with m number of atoms, after "removing", the number of loops is reduced from $2^{m+n-4} + 2^n + 2^m - 2$ to $2^n + 2^m - 2$.

DLP and Strongly Splitting

A set U of atoms is called a *splitting set* of a program P, if for each $r \in P$, $head(r) \cap U \neq \emptyset$ implies $Atoms(r) \subseteq U$.

The *bottom* and *top* of P *w.r.t.* U are denoted as $b_U(P)$ and $t_U(P)$. And they are defined as : $b_U(P) = \{r \in P \mid head(r) \cap U \neq \emptyset\}$. $t_U(P) = P \setminus b_U(P)$.

And $e_U(P, X)$ is the set of rules obtained by deleting some rules from P *w.r.t.* U and X.

A *solution* of P *w.r.t.* U is a pair $\langle X, Y \rangle$ of sets of atoms *s.t.*

- X is an answer set of $b_U(P)$,
- Y is an answer set of $e_U(P \setminus b_U(P), X)$.

Example 2. Consider the logic program P_2:

$$a \leftarrow not\, d. \;\; d \leftarrow not\, c. \;\; a \leftarrow c, d. \;\; c \leftarrow .$$

$\{c, d\}$ is a splitting set of P_2 and the bottom of P *w.r.t.* $\{c, d\}$ is $b_{\{c,d\}}(P_2) = \{d \leftarrow not\, c.\ c \leftarrow .\}$. Further more, $e_{\{c,d\}}(P_2 \setminus b_{\{c,d\}}(P_2), \{c\}) = \{a \leftarrow .\}$. And $\langle \{c\}, \{a\} \rangle$ is a solution of P_2 *w.r.t.* $\{c, d\}$.

Theorem 2 (Splitting Set Theorem). Let U be a splitting set of a program P. A set S is an answer set of P iff $S = X \cup Y$ for some solution $\langle X, Y \rangle$ of P *w.r.t.* U.

Program Simplification

A rule r is called an *external support* of a loop L if $head(r) \in L$ and $L \cap body^+(r) = \emptyset$. Let $R^-(L)$ be the set of external support rules of L.

A loop L of a logic program P is called *proper* if there does not exist another loop L' of P s.t.

- $L' \subset L$ and $R^-(L') \subseteq R^-(L)$, or
- $R^-(L') \neq \emptyset$ and $R^-(L') \subset R^-(L)$.

Proposition 5. Let P be a logic program and L a loop of P. If L is a proper loop of P, then L is an elementary loop of P, but not vice versa.

Example 1 (Continued). Program P_1 has three proper loops: $\{q\}$, $\{r, q\}$ and $\{p, r, q\}$.

$\{p, r\}$ and $\{p\}$ are not proper loops as $R^-(\{p, r\}) = \{p \leftarrow ., r \leftarrow q.\}$, $R^-(\{p\}) = \{p \leftarrow ., p \leftarrow r.\}$ and $R^-(\{p, r, q\}) = \{p \leftarrow .\}$.

$\{r\}$ is not a proper loops as $R^-(\{r\}) = \{r \leftarrow p., r \leftarrow q.\}$ and $R^-(\{q, r\}) = \{r \leftarrow p.\}$.

A Future Direction

The idea of splitting will find many other uses, for instance in the investigation of incremental ASP solvers and forgetting.

References

[1] Lifschitz, V., and Turner, H. 1994. Splitting a logic program.. In *Proceedings of the 11th International Conference on Logic Programming(ICLP-94)* 23-37.

Source Code

For more information and source code, please visit:

`http://ss.sysu.edu.cn/~wh/splitting.html`