viasp: A visualization tool for Answer Set Programming (Extended Abstract)

Luis Glaser, Javier Romero, Torsten Schaub, and Stephan Zwicknagl

University of Potsdam, Germany

The idea of Answer Set Programming (ASP; Lifschitz 2008) is to describe a problem by a logic program whose answer sets correspond to the solutions of the problem, and to use an ASP solver to compute them. In practice, exploring these answer sets can be difficult: they have no internal structure, may contain many atoms, and can be numerous. This poses a challenge for users learning ASP, as well as for experts trying to understand the solutions of a problem or to debug an encoding. To address this, we present *viasp*,[1] an interactive tool for visualizing the answer sets of a logic program. Its goal is to provide a simple yet structured view of the answer sets, allowing the user to navigate within and between them with ease.

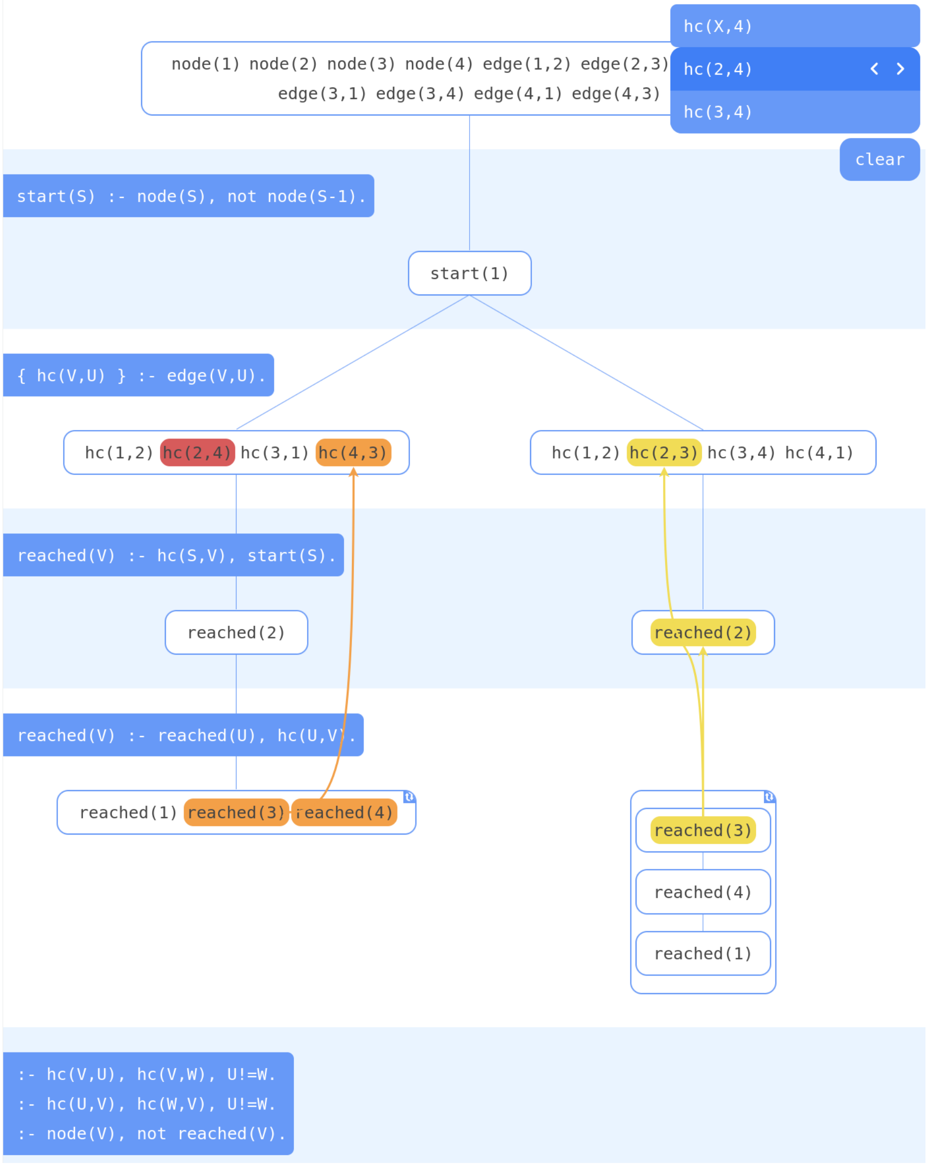
As an example, consider a logic program `hamiltonian.lp` for the Hamiltonian cycle problem in a given graph. The command `viasp hamiltonian.lp 0` computes all answer sets using *clingo* (Gebser et al. 2019) and opens the visualization of Figure 1 in the browser.

viasp represents each answer set as the result of successively applying the rules of the program. The rules are arranged in horizontal blocks, ordered according to their dependencies. They appear on the left, while facts are shown at the top. Answer sets are presented in a common tree. Our example program has two of them. They share the facts at the root and the node containing `start(1)` below it. The tree diverges in the next block, where each answer set chooses different `hc/2` atoms. The following blocks show the derivation of the `reached/1` atoms. Both answer sets derive the same atoms, but follow different paths. The integrity constraints are grouped at the bottom, as they do not contribute any atoms.

viasp also provides interactive functionality. The nodes at the second rule for `reached/1` derive their atoms through many applications of the rule. This is indicated by an additional button in their top-right corner. In the figure, we have clicked this button on the right node, which then displays the sequence of applications of the rule. Next, we clicked on the atoms `reached(3)` of both answer sets, and now the coloured arrows show which atoms were used to derive each of them. We also entered the atom `hc(X,4)` in the top-right query box. *viasp* responded by showing the two ground instances that appear in the answer sets, `hc(2,4)` and `hc(3,4)`. Clicking on the first one highlighted it in red.

Additional features of *viasp* include the ability to contract and expand nodes containing many atoms, a native zoom function for navigating through several answer sets, and an option to reorder the rules while respecting their dependencies.

[1] Available at `https://github.com/potassco/viasp`.



```
start(S) :- node(S), not node(S-1).
{ hc(V,U) } :- edge(V,U).
reached(V) :- hc(S,V), start(S).
reached(V) :- reached(U), hc(U,V).
:- hc(V,U), hc(V,W), U!=W.
:- hc(U,V), hc(W,V), U!=W.
:- node(V), not reached(V).
```

hc(X,4)
hc(2,4)
hc(3,4)
clear
node(1) node(2) node(3) node(4) edge(1,2) edge(2,3)
edge(3,1) edge(3,4) edge(4,1) edge(4,3)
start(1)
hc(1,2) hc(2,4) hc(3,1) hc(4,3)
hc(1,2) hc(2,3) hc(3,4) hc(4,1)
reached(2)
reached(2)
reached(1) reached(3) reached(4)
reached(3)
reached(4)
reached(1)

Fig. 1. Visualization of the Hamiltonian cycle problem in *viasp*.

Acknowledgments. This work was supported by the DFG grant SCHA 550/15 (Germany).

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

Gebser, M., R. Kaminski, B. Kaufmann, and T. Schaub (2019). "Multi-shot ASP solving with clingo". In: *Theory and Practice of Logic Programming* 19.1, pp. 27–82. DOI: 10.1017/S1471068418000054.

Lifschitz, V. (2008). "What Is Answer Set Programming?" In: *Proceedings of the Twenty-third National Conference on Artificial Intelligence (AAAI'08)*. Ed. by D. Fox and C. Gomes. AAAI Press, pp. 1594–1597.