Definition of Safety

Günther Wullaert

May 2022

1 Language

1.1 Term and Pools

We inductively define terms as

- all numerals, symbolic constants, and variables are terms
- $f(\boldsymbol{t})$ is a term, if f is a symbolic constant and $\boldsymbol{t}$ is a pool
- $(t_1 \star t_2)$ is a term, if t_1 and t_2 are terms and $\star$ is one of the symbols (+ - $\times$ / ..)
- $\langle \boldsymbol{t} \rangle$ is a term, if $\boldsymbol{t}$ is a pool, which can have a possibly empty set of terms.

A tuple of terms has the following form $t_1, ..., t_n$ where t_i is a term.
A pool is an expression of the form $\dot{t_1}; ...; \dot{t_n}$ where $n \geq 1$ and each $\dot{t_i}$ is a tuple of terms. In particular, every tuple of terms is a pool.

1.2 Constants

We inductively define a term to be *constant* if:

- it is a numeral
- it has form $t \star u$ where t and u are *constant* and $\star$ is one of the symbols (+ - $\times$ /)

1.3 Atoms and Literals

An atom has form $p(\boldsymbol{t})$ where p is a predicate symbol and $\boldsymbol{t}$ is a pool.
A literal is either an atom or atom preceded by not.

2 Safety

We define the function *provide*() to check if a statement is safe. The provide function returns a set of tuples $\{\dot{t_1}, ..., \dot{t_n}\}$. Each tuple $\dot{t_i}$ has the form (p, d), where p is a set of variables the statement provides if d variables are provided. If 2 tuples share the same d, the p can be merged. If 2 tuples share the same p, the d can be merged.
For example:

$$\{(\{X\}, \{\}), (\{Y\}, \{\})\} = \{(\{X, Y\}, \{\})\} \tag{1}$$

$$\{(\{\}, \{X\}), (\{\}, \{Y\})\} = \{(\{\}, \{X, Y\})\} \tag{2}$$

2.1 Helper Functions

The *depend* function takes a set of tupels $\{(p_1, d_1), ..., (p_n, d_n)\}$ and sets $d_i = p_i \cup d_i$ and $p_i = \emptyset$ for each tuple (p_i, d_i)
For Example:

$$\begin{aligned} depend(\{(\{X\}, \{\}), (\{Y\}, \{Z\})\}) &= \{(\{\}, \{X\}), (\{\}, \{Y, Z\})\} \\ &= \{(\{\}, \{X, Y, Z\})\} \end{aligned} \tag{3}$$

The *merge* function takes a set of tupels $\{(p_1, d_1), ..., (p_n, d_n)\}$ and a set of variables v
For Example:

$$merge(\{(\{X\}, \{\})\}, \{Y\}) = \{(\{X\}, \{Y\})\} \tag{4}$$

The output of the *depend* function will also always be in the form $\{(\{\}, d)\}$ where d will contain all the variables. The output of the depend function can thus easily be converted to the input parameter v of the *merge* function.
For Example:

$$\begin{aligned} &merge(\{(\{X\}, \{\})\}, depend(\{(\{Y\}, \{\})\})) \\ &\quad = merge(\{(\{X\}, \{\})\}, \{Y\}) \\ &\quad = \{(\{X\}, \{Y\})\} \end{aligned} \tag{5}$$

2.2 Terms

2.2.1 Constants

For any numeral n and symbolic constant f:

$$provide(n) = provide(f) = \emptyset \tag{6}$$

2.2.2 Variables

For any variable X:

$$provide(X) = \{(\{X\}, \{\})\} \tag{7}$$

2.2.3 Tuples

For any tuple of terms $t_1, ..., t_n$:

$$provide(t_1, ..., t_n) = provide(t_1) \cup ... \cup provide(t_n) \tag{8}$$

For Example:

$$\begin{aligned} provide(X, Y) &= provide(X) \cup provide(Y) \\ &= \{(\{X\}, \{\})\} \cup \{(\{Y\}, \{\})\} \\ &= \{(\{X\}, \{\}), (\{Y\}, \{\})\} \\ &= \{(\{X, Y\}, \{\})\} \end{aligned} \tag{9}$$

2.2.4 Pools

For any pool of terms $\dot{t_1}; ...; \dot{t_n}$:

$$provide(\dot{t_1}; ...; \dot{t_n}) = provide(\dot{t_1}) \cap ... \cap provide(\dot{t_n}) \tag{10}$$

For Example:

$$\begin{aligned} provide(X, Y; Y) &= provide(X, Y) \cap provide(Y) \\ &= provide(X, Y) \cap \{(\{Y\}, \{\})\} \\ &= (provide(X) \cup provide(Y)) \cap \{(\{Y\}, \{\})\} \\ &= (\{(\{X\}, \{\})\} \cup \{(\{Y\}, \{\})\}) \cap \{(\{Y\}, \{\})\} \\ &= \{(\{X, Y\}, \{\})\} \cap \{(\{Y\}, \{\})\} \\ &= \{(\{Y\}, \{\})\} \end{aligned} \tag{11}$$

2.2.5 Terms

For a term of form $f(\boldsymbol{t})$, where f a symbolic constant and $\boldsymbol{t}$ a pool:

$$provide(f(\boldsymbol{t})) = provide(\boldsymbol{t}) \tag{12}$$

For Example:

$$\begin{aligned} provide(f(X)) &= provide(X) \\ &= \{(\{X\}, \{\})\} \end{aligned} \tag{13}$$

For a term of form $t_1 \star t_2$, where t_1 and t_2 are terms which are not *constants* and $\star$ is one of the symbols (+ - × / ..):

$$provide(t_1 \star t_2) = depend(provide(t_1) \cup provide(t_2)) \tag{14}$$

For Example:

$$\begin{aligned} provide(X * Y) &= depend(provide(X) \cup provide(Y)) \\ &= depend(\{(\{X\}, \{\})\} \cup \{(\{Y\}, \{\})\}) \\ &= depend(\{(\{X, Y\}, \{\})\}) \\ &= \{(\{\}, \{X, Y\})\} \end{aligned} \tag{15}$$

$$\begin{aligned} provide((X * Y) + Z) &= depend(provide(X * Y) \cup provide(Z)) \\ &= depend(\{(\{\}, \{X, Y\})\} \cup \{(\{Z\}, \{\})\}) \\ &= depend(\{(\{Z\}, \{X, Y\})\}) \\ &= \{(\{\}, \{X, Y, Z\})\} \end{aligned} \tag{16}$$

For a term of form $t \star c$, where c is a *constant* and t is a term the following holds for $\star$ being one of the symbols (+ - x):

$$provide(t \star c) = provide(c \star t) = provide(t) \tag{17}$$

For Example:

$$\begin{aligned} provide(X + 1) &= provide(X) \\ &= \{(\{X\}, \{\})\} \end{aligned} \tag{18}$$

For a term of form $-t$, where t is a term:

$$provide(-t) = provide(t) \tag{19}$$

For Example:

$$\begin{aligned} provide(-X) &= provide(X) \\ &= \{(\{X\}, \{\})\} \end{aligned} \tag{20}$$

2.3 Atoms and Literals

2.3.1 Atoms

For an atom of form $p(\boldsymbol{t})$, where $\boldsymbol{t}$ is a pool:

$$provide(p(\boldsymbol{t})) = provide(\boldsymbol{t}) \tag{21}$$

For Example:

$$\begin{aligned} provide(p(X; Y)) &= provide(X; Y) \\ &= provide(X) \cap provide(Y) \\ &= \{(\{X\}, \{\})\} \cap \{(\{Y\}, \{\})\} \\ &= \emptyset \end{aligned} \tag{22}$$

2.3.2 Literals

For an literal of form $not \quad a$, where a is an atom:

$$provide(not \quad a) = depend(provide(a)) \tag{23}$$

For Example:

$$\begin{aligned} provide(not \quad p(X)) &= depend(provide(p(X))) \\ &= depend(provide(X)) \\ &= depend(\{(\{X\}, \{\})\}) \\ &= \{(\{\}, \{X\})\} \end{aligned} \tag{24}$$

For an literal of form a, where a is an atom:

$$provide(a) = provide(a) \tag{25}$$

For an literal of form $not \quad l$, where l is another literal:

$$provide(not \quad l) = depend(provide(l)) \tag{26}$$

2.4 Comparisons

2.4.1 Comparisons

For an comparison of form $t_1 \star t_2$, where t_1 and t_2 are terms and $\star$ is one of the symbols $(\leq, \geq, <, >, \neq)$:

$$provide(t_1 \star t_2) = depend(provide(t_1) \cup provide(t_2)) \tag{27}$$

For Example:

$$\begin{aligned} provide(X \geq Y) &= depend(provide(X) \cup provide(Y)) \\ &= depend(\{(\{X\}, \{\})\} \cup \{(\{Y\}, \{\})\}) \\ &= depend(\{(\{X, Y\}, \{\})\}) \\ &= \{(\{\}, \{X, Y\})\} \end{aligned} \tag{28}$$

2.4.2 Assignments

For an comparison of form $t_1 = t_2$, where t_1 and t_2 are terms:

$$\begin{aligned} provide(t_1 = t_2) = {} & merge(provide(t_1), depend(provide(t_2))) \\ & \cup merge(provide(t_2), depend(provide(t_1))) \end{aligned} \tag{29}$$

For Example:

$$\begin{aligned} provide(X = Y) &= merge(provide(X), depend(provide(Y))) \\ &\quad \cup merge(provide(Y), depend(provide(X))) \\ &= merge(\{(\{X\}, \{\})\}, depend(\{(\{Y\}, \{\})\})) \\ &\quad \cup merge(\{(\{Y\}, \{\})\}, depend(\{(\{X\}, \{\})\})) \\ &= merge(\{(\{X\}, \{\})\}, Y) \cup merge(\{(\{Y\}, \{\})\}, X) \\ &= \{(\{X\}, \{Y\})\} \cup \{(\{Y\}, \{X\})\} \\ &= \{(\{X\}, \{Y\}), (\{Y\}, \{X\})\} \end{aligned} \tag{30}$$

Hypothetical Example (I don't know how to create this situation in clingo):

$$\begin{aligned}
&provide(\{(\{X\},\{\}),(\{\},\{Y\})\} = \{(\{Z\},\{\}),(\{T\},\{W\})\}) \\
&\quad = merge(\{(\{X\},\{\}),(\{\},\{Y\})\}, depend(\{(\{Z\},\{\}),(\{T\},\{W\})\})) \\
&\qquad \cup\, merge(\{(\{Z\},\{\}),(\{T\},\{W\})\}, depend(\{(\{X\},\{\}),(\{\},\{Y\})\})) \\
&\quad = merge(\{(\{X\},\{\}),(\{\},\{Y\})\}, \{T,W,Z\}) \\
&\qquad \cup\, merge(\{(\{Z\},\{\}),(\{T\},\{W\})\}, \{X,Y\}) \\
&\quad = \{(\{X\},\{T,W,Z\}),(\{\},\{T,W,Y,Z\})\} \cup \{(\{Z\},\{X,Y\}),(\{T\},\{W,X,Y\})\} \\
&\quad = \{(\{X\},\{T,W,Z\}),(\{\},\{T,W,Y,Z\}),(\{Z\},\{X,Y\}),(\{T\},\{W,X,Y\})\}
\end{aligned} \tag{31}$$

2.5 Rule

For a rule r in the form of

$$H_1 \vee ... \vee H_k \leftarrow B_1 \wedge ... \wedge B_m \tag{32}$$

The following holds:

$$\begin{aligned}
provide(r) = {} & depend(provide(H_1)) \cup ... \cup depend(provide(H_k)) \\
& \cup\, provide(B_1) \cup ... \cup provide(B_m)
\end{aligned} \tag{33}$$

For Example:

$$\begin{aligned}
provide(a(X) \leftarrow b(X)) &= depend(provide(a(X))) \cup provide(b(X)) \\
&= depend(provide(X)) \cup provide(X) \\
&= depend(\{(\{X\},\{\})\}) \cup \{(\{X\},\{\})\} \\
&= \{(\{\},\{X\})\} \cup \{(\{X\},\{\})\} \\
&= \{(\{\},\{X\}),(\{X\},\{\})\}
\end{aligned} \tag{34}$$

3 Other Examples

$$\begin{aligned}
provide(p(X, Y+Y)) &= provide(X, Y+Y) \\
&= provide(X) \cup provide(Y+Y) \\
&= \{(\{X\},\{\})\} \cup depend(provide(Y) \cup provide(Y)) \\
&= \{(\{X\},\{\})\} \cup depend(\{(\{Y\},\{\})\} \cup \{(\{Y\},\{\})\}) \\
&= \{(\{X\},\{\})\} \cup depend(\{(\{Y\},\{\})\}) \\
&= \{(\{X\},\{\})\} \cup \{(\{\},\{Y\})\} \\
&= \{(\{X\},\{\}),(\{\},\{Y\})\}
\end{aligned} \tag{35}$$