Report: Argument Builder v2

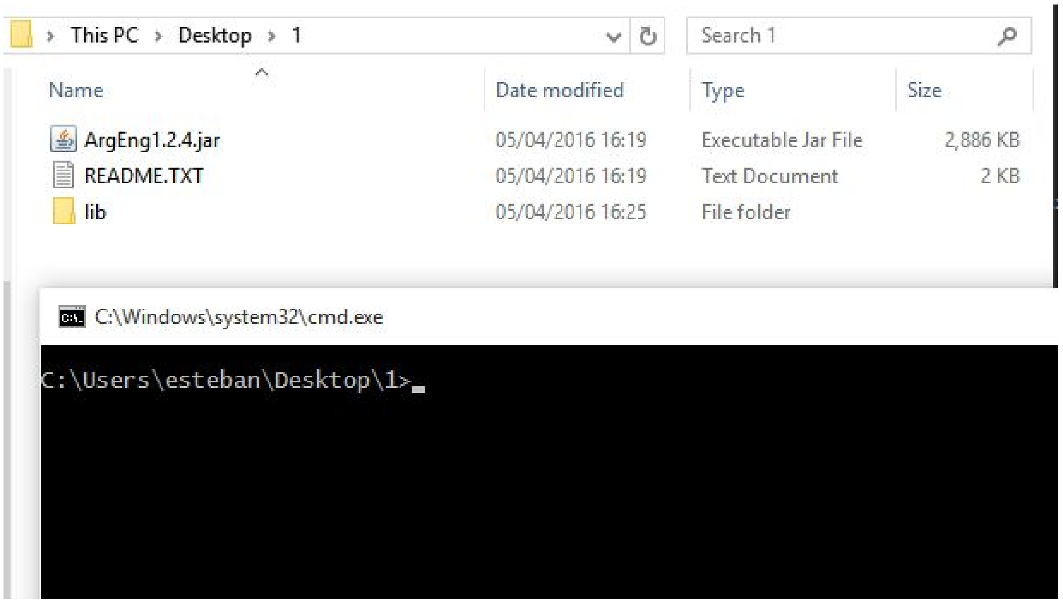
Author: Esteban Guerrero

Contact: esteban.guerrero@umu.se

Abstract: A new version of the Argument Builder engine is presented. This version of the builder is based on a Well-Founded Semantics evaluation using the DLV System.

Quick start:

1. The argument builder is a self contained jar file (current version 1.2.4) which requires external libraries provided in the folder "lib".
2. Expand the compress file ArgEng.zip
3. For using as a standalone tool:
    a. Open a command line window in the folder of the ArgEng file (tip in Windows: Shift + Right click in the folder → "Open command window here")



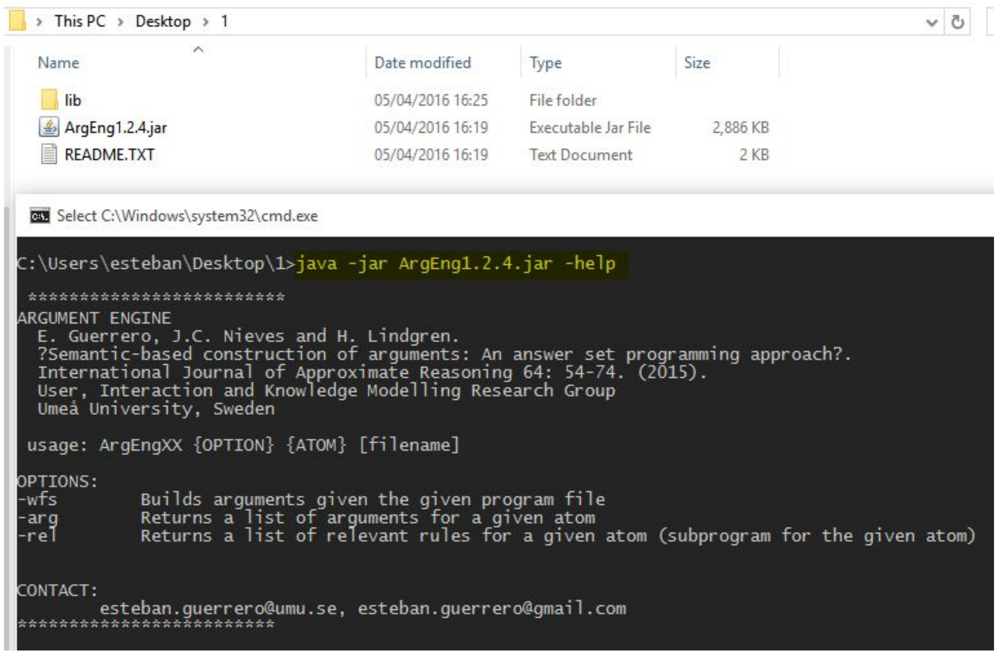
This PC › Desktop › 1

Name	Date modified	Type	Size
ArgEng1.2.4.jar	05/04/2016 16:19	Executable Jar File	2,886 KB
README.TXT	05/04/2016 16:19	Text Document	2 KB
lib	05/04/2016 16:25	File folder	

C:\Windows\system32\cmd.exe

```
C:\Users\esteban\Desktop\1>_
```

    b. For help of the Argument Builder usage use the command **-help**, such as: java -jar ArgEng1.2.4.jar -help

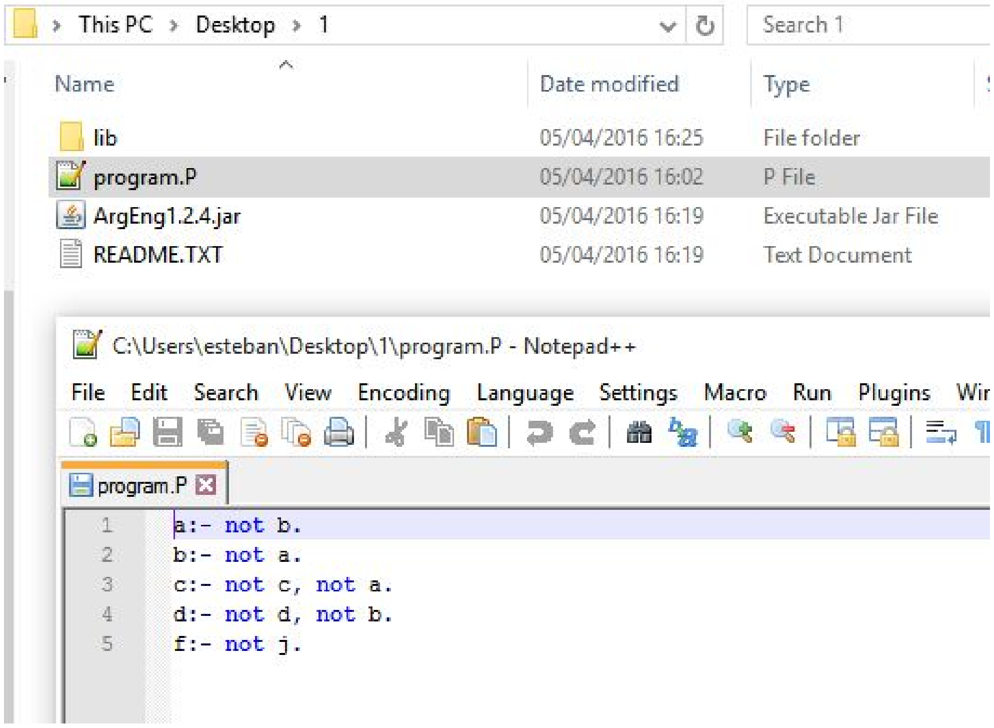


Name	Date modified	Type	Size
lib	05/04/2016 16:25	File folder	
ArgEng1.2.4.jar	05/04/2016 16:19	Executable Jar File	2,886 KB
README.TXT	05/04/2016 16:19	Text Document	2 KB

Select C:\Windows\system32\cmd.exe

```
C:\Users\esteban\Desktop\1>java -jar ArgEng1.2.4.jar -help

 *************************
ARGUMENT ENGINE
  E. Guerrero, J.C. Nieves and H. Lindgren.
  ?Semantic-based construction of arguments: An answer set programming approach?.
  International Journal of Approximate Reasoning 64: 54-74. (2015).
  User, Interaction and Knowledge Modelling Research Group
  Umeå University, Sweden

 usage: ArgEngXX {OPTION} {ATOM} [filename]

OPTIONS:
-wfs        Builds arguments given the given program file
-arg        Returns a list of arguments for a given atom
-rel        Returns a list of relevant rules for a given atom (subprogram for the given atom)


CONTACT:
        esteban.guerrero@umu.se, esteban.guerrero@gmail.com
*************************
```

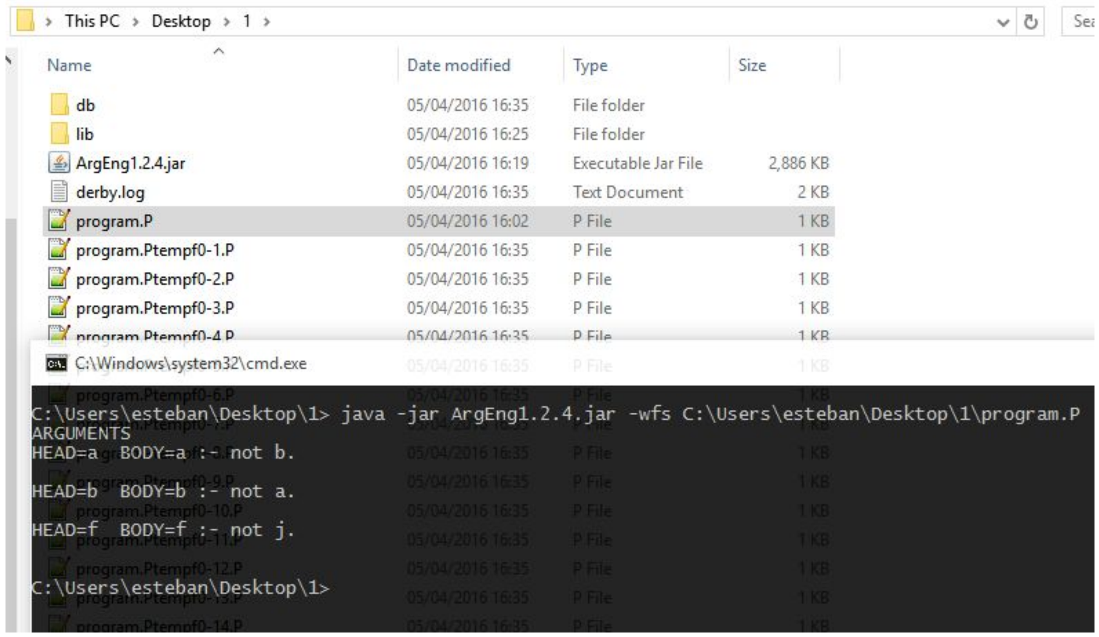
c. Create a logic program in a file using Prolog notation, for instance:



Name	Date modified	Type
lib	05/04/2016 16:25	File folder
program.P	05/04/2016 16:02	P File
ArgEng1.2.4.jar	05/04/2016 16:19	Executable Jar File
README.TXT	05/04/2016 16:19	Text Document

```prolog
a:- not b.
b:- not a.
c:- not c, not a.
d:- not d, not b.
f:- not j.
```

d. Building arguments for “program.P” by using the **-wfs** option and passing the location of the “program.P” file, such as: **java -jar ArgEng1.2.4.jar -wfs C:\Users\esteban\Desktop\1\program.P**



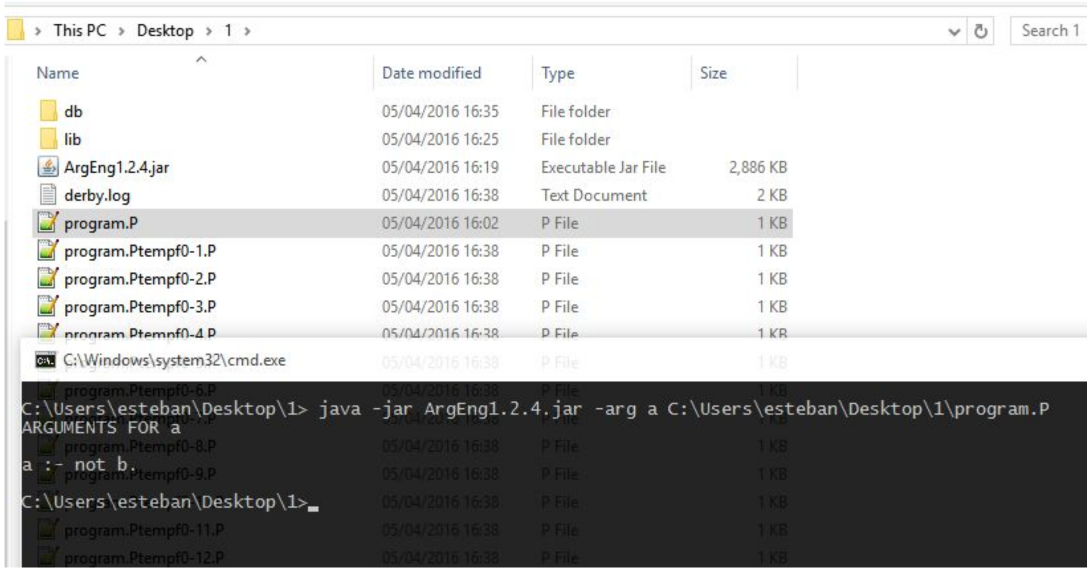
This PC › Desktop › 1 ›

Name	Date modified	Type	Size
db	05/04/2016 16:35	File folder	
lib	05/04/2016 16:25	File folder	
ArgEng1.2.4.jar	05/04/2016 16:19	Executable Jar File	2,886 KB
derby.log	05/04/2016 16:35	Text Document	2 KB
program.P	05/04/2016 16:02	P File	1 KB
program.Ptempf0-1.P	05/04/2016 16:35	P File	1 KB
program.Ptempf0-2.P	05/04/2016 16:35	P File	1 KB
program.Ptempf0-3.P	05/04/2016 16:35	P File	1 KB

C:\Windows\system32\cmd.exe

```
C:\Users\esteban\Desktop\1> java -jar ArgEng1.2.4.jar -wfs C:\Users\esteban\Desktop\1\program.P
ARGUMENTS
HEAD=a  BODY=a :- not b.

HEAD=b  BODY=b :- not a.

HEAD=f  BODY=f :- not j.

C:\Users\esteban\Desktop\1>
```

e. Retrieving the set of arguments for a given atom using the option: **-arg**, for instance retrieving the set of arguments (if any) for the atom ”a” in the program “program.P”: **java -jar ArgEng1.2.4.jar -arg a C:\Users\esteban\Desktop\1\program.P**



This PC › Desktop › 1 ›

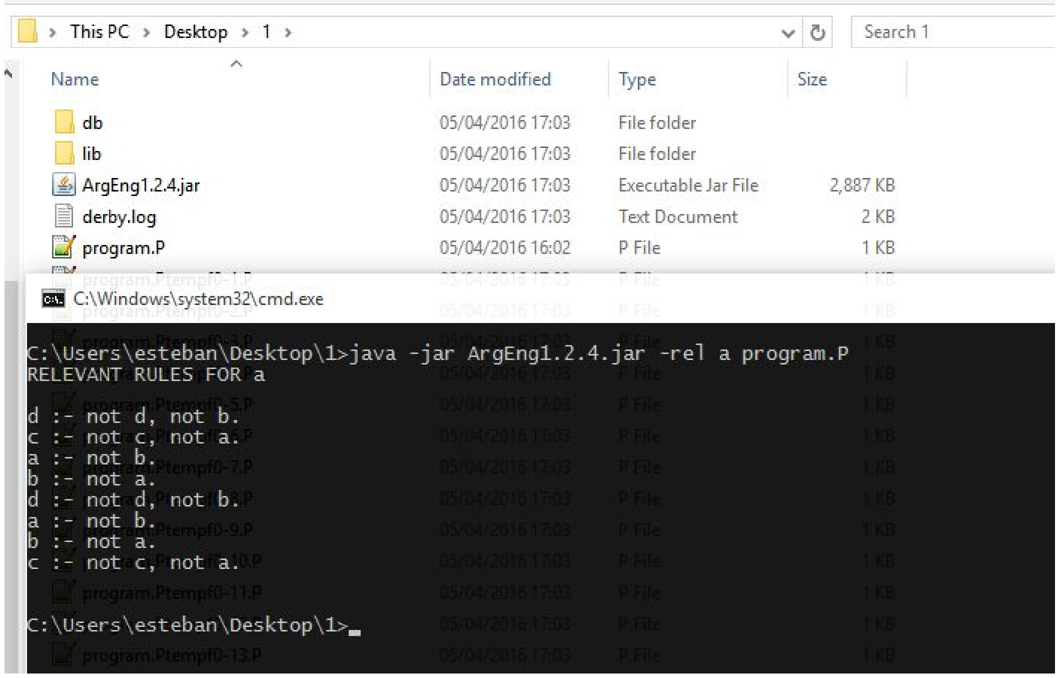
Name	Date modified	Type	Size
db	05/04/2016 16:35	File folder	
lib	05/04/2016 16:25	File folder	
ArgEng1.2.4.jar	05/04/2016 16:19	Executable Jar File	2,886 KB
derby.log	05/04/2016 16:38	Text Document	2 KB
program.P	05/04/2016 16:02	P File	1 KB
program.Ptempf0-1.P	05/04/2016 16:38	P File	1 KB
program.Ptempf0-2.P	05/04/2016 16:38	P File	1 KB
program.Ptempf0-3.P	05/04/2016 16:38	P File	1 KB

C:\Windows\system32\cmd.exe

```
C:\Users\esteban\Desktop\1> java -jar ArgEng1.2.4.jar -arg a C:\Users\esteban\Desktop\1\program.P
ARGUMENTS FOR a

a :- not b.

C:\Users\esteban\Desktop\1>_
```

f. Retrieving the set of *relevant rules* for a given atom using the option: **-rel**, for instance retrieving the set of rules (if any) for the atom ”a” in the program

“program.P”: **java -jar ArgEng1.2.4.jar -rel a C:\Users\esteban\Desktop\1\program.P**



This PC > Desktop > 1 >
Search 1
Name
Date modified
Type
Size
db 05/04/2016 17:03 File folder
lib 05/04/2016 17:03 File folder
ArgEng1.2.4.jar 05/04/2016 17:03 Executable Jar File 2,887 KB
derby.log 05/04/2016 17:03 Text Document 2 KB
program.P 05/04/2016 16:02 P File 1 KB
C:\Windows\system32\cmd.exe

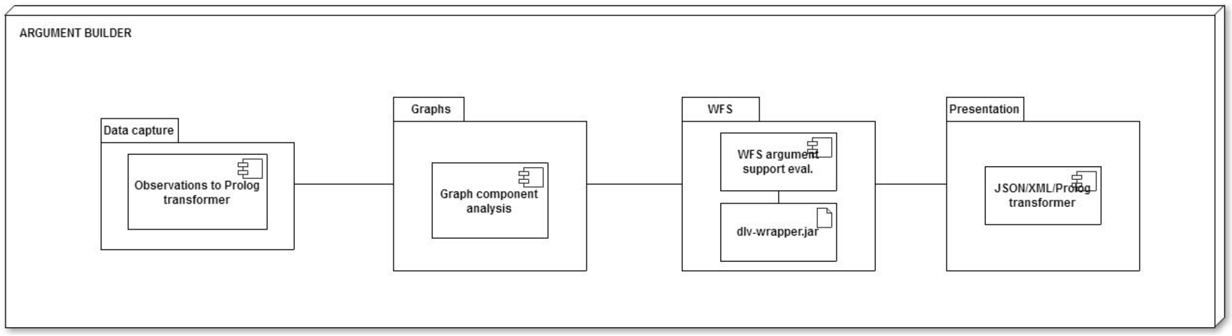
```
C:\Users\esteban\Desktop\1>java -jar ArgEng1.2.4.jar -rel a program.P
RELEVANT RULES FOR a

d :- not d, not b.
c :- not c, not a.
a :- not b.
b :- not a.
d :- not d, not b.
a :- not b.
b :- not a.
c :- not c, not a.

C:\Users\esteban\Desktop\1>
```

Notes:

- The program accept negation as failure (NAF) as well as strong negation, e.g.: the program

  P:={-b:- not a. c:- not c, not a.
  f:- not j.
  }
- Some errors can be generated when blank lines are left in the program file.
- The argument builder generates some temporal files which are automatically deleted by the program. Those files are subprograms of the original program which are generated when the relevant rules are obtained. More details in the paper. *E. Guerrero, J.C. Nieves and H. Lindgren. “Semantic-based construction of arguments: An answer set programming approach”. International Journal of Approximate Reasoning 64: 54-74. (2015).*

Architecture:



ARGUMENT BUILDER
Data capture
Observations to Prolog transformer
Graphs
Graph component analysis
WFS
WFS argument support eval.
dlv-wrapper.jar
Presentation
JSON/XML/Prolog transformer