**Spinoza Explosion-Users Guide**

**Introduction**

The logical flow of Spinoza’s *Ethics* can be represented by a mathematical object called a digraph. The mathematical definition of a digraph is a set of vertices with an asymmetric relationships on the vertices called arcs. Since humans are visually oriented, the logical flow of the *Ethics* can be better understood if it is depicted graphically. The vertices represent the definitions, axioms, propositions, etc. and the arcs denote the non-reflexive relationship “necessary for the demonstration of.” For example, both 1D3 and 1P1 are logically necessary for the demonstration of 1P5. There are several examples of graphical representations of the five Parts of the *Ethics* on the Web, but because of the large number of elements and the complexity of their interactions using these representations can be inconvenient. (There are representations of Parts 1-3 on my website.)

*Spinoza\_Explosion* is a Python 2.7 script that can be used interactivelyto create representations focused on single elements rather than whole parts of the *Ethics.* There are two types of “explosions:” forward and backward. “Forward” explodes a selected element into *all* propositions directly or indirectly connected to the element. “Backward” explodes a selected element into *all* digraph elements (propositions, definitions, axioms, etc) directly or indirectly necessary for the proof of the selected element. Forward explosions generate all results stemming from an element, and backward explosions depict all elements required for the logical proof of an element. Examples of forward and backward explosions, both of which are cutoff to save space, are given on page 3.

**Python Scripts**

Spinoza’s logic for the *Ethics* is specified in a Python module called *final\_props.py*. This module was developed from manually developed text files that associate each element of the *Ethics* with the elements that immediately proceed it logically. The proceeding elements are simply those specified by Spinoza as documented in the Elwes translation. A Python script was written to invert Spinoza’s specifications, that is to associate each element with its immediately succeeding elements. The inverted specification, which is logically equivalent to Spinoza’s specification, is also stored in *final\_props.py*.

Translating the *Ethics* into a digraph that could be processed by a Python script required a few changes to the Elwes translation which are necessary to keep in mind when using the system:

1. Arabic numerals were substituted for Roman numerals and elements are numbered as PEN,

where P is the Part of the *Ethics*, E is the element type (D=Definition, A=Axiom, P=Proposition, F=aFfect, L=Lemma, T=posTulate) and N is the number assigned by Spinoza. For example, Part 3 proposition 27 is designated as 3P27. Note that capitals letters are required for element types.

1. Corollaries are numbered with a decimal point. For example Part 3, proposition 27, Corollary 2 is designated 3P27.2. In all cases the proposition associated with the corollary is assumed to be logically necessary fo its proof even when not explicitly stated by Spinoza.
2. The unnumbered axioms in Part 2 are numbered sequentially as they appear from 2A6 to 2A9. The unnumbered definition in Part 2 is numbered 2D8.
3. The general definition of affect is used several times and is numbered 3F0.
4. All elements that were not connected by Spinoza to another element have been omitted from the digraph. There are 35 such elements.
5. Alternate proofs are omitted.

Processing is controlled by a Python 2.7 script*, Spinoza\_Explosion*.py , which imports *final\_props.py,* a network analysis system called *igraph* and a plotting software named *cairo*. Both *igraph* and *cairo* are documented on the web. *Spinoza\_Explosion.py* and *final\_props.py* are open source which can be obtained from the author by requesting them at hroseman@gmail.com.

**Implementation**

1. Navigate to the *Spinoza Project* on the website *sites.google.com/site/hroseman* and download the msi file *Spin\_Explode-0.1-win32.msi*.
2. Click on the msi and follow the instructions. You will be warned that the file can harm your computer. You will have to ignore this warning if you wish to use the program. As a reader of and believer in 4P18S I have no intention of harming anyone’s computer.
3. *Spinoza\_Explosion.exe* will be in the folder *C:/Program Files (x86)/Spin\_Explode.* It will be marked by an icon that looks like an explosion. If you wish you can use the icon to make a shortcut for your desktop. To start the program click on the exec or the shortcut and you will see the following:



1. Operation should be self-evident. You may wish to reset the “MAXIMUM NUMBER OF ELEMENTS DISPLAYED” depending on your requirements. There are 408 elements in the complete digraph so the presentations can become difficult to use if too many are displayed. The default is set to 100 elements.
2. This is the first product of my Spinoza project. I expect that there will be a very small number of errors in the data and there will probably be suggestions for improving *Spinoza\_Explosion*. Address all corrections, suggestions and questions to *hroseman@gmail.com*. I will send the Python source upon request, but I wish to control modification of the basic data files so that the Spinoza community will have access to one data file that will eventually become error free.

Backward Explosion of 1P25



Forward Explosion of 4P25

*Color scheme: element analyzed-lime; definitions-grey; axioms-pink; propositions-yellow; leafs-amber; affects-aqua; lemmas-turquoise; postulates-magenta*