FAQ: “Sessie” Introduction

The toy “universe” is a string of characters, whose initial state can have any length and can contain any characters. (There are two infinities here: the length of the state string and the size of the alphabet from which the characters are taken: both have no imposed bound.) The “laws of nature” for the SSS are codified by a ruleset: a set of replacement instructions, each giving a string to search for and the string to replace it with. Ex: {“AB”→“BA”}. (Now we have more infinities: there is no limit on the number of rules in the ruleset, nor on the size of any of the strings, nor on the characters in the strings.)

In a sequential substitution system, rules must be applied sequentially in order (use the first rule if possible, the second only if the first fails, etc.) from left-to-right: if the first usable rule can be used at multiple locations in the state string, we apply it in the first possible (left-most) position.

Each application of any rule is an event, and each event destroys and/or creates cells (or letters) in the state string. Two events are considered to be causally connected if a cell created by one is destroyed by the other. (Later we’ll explain more about how the causal network connections work, i.e. the lines that connect the event nodes.)

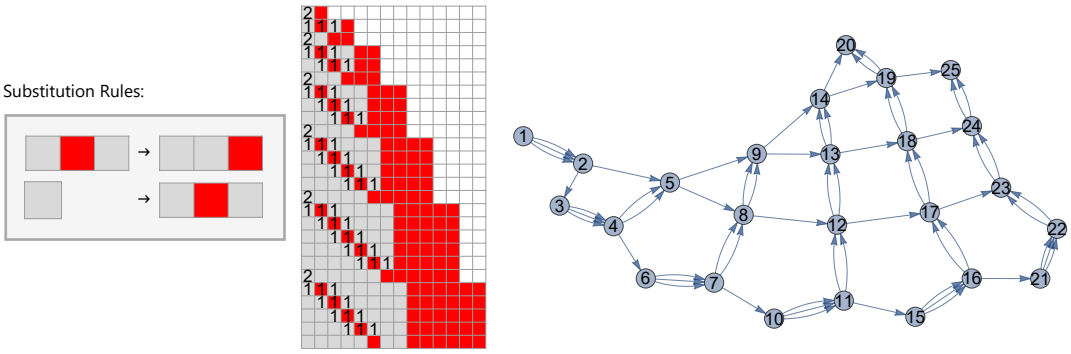
The causal network of an SSS can be visualized by treating all events as graph nodes and the causal connections between them as graph edges.

The basic command to display a Sessie is SSS. The template for usage is:

SSS[ruleset, initial state, number of steps, options]

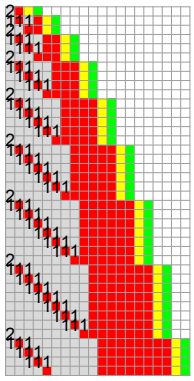
Ex:

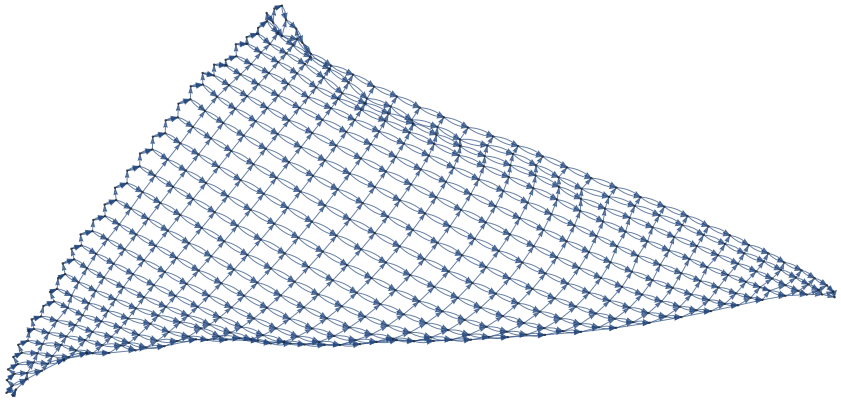
```
In[ ]:= sss1 = SSS[{"ABA" → "AAB", "A" → "ABA"},
    "AB", 25, RulePlacement → Left, SSSSize → 100, NetSize → 300];
```



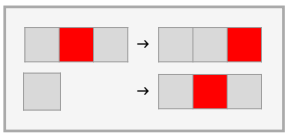
Try changing the rules in the ruleset, or the initial state string, or the number of steps. Or try it with some options. Some common options include changing the size or maximum event/node shown in the SSS or the Net, or changing how VertexLabels are displayed:

In[ ]:=
```
sss2 = SSS[{"ABA" → "AAB", "A" → "ABA"}, "ABCD",
    500, SSSMax → 40, SSSSize → 100, NetSize → 450, VertexLabels → None];
```





Substitution Rules:



To find more information on any function use a question mark. For example:

In[ ]:=
```
? SSS
```

Out[ ]=

Symbol

SSS[*ruleset*, *init*, *n*, *opts*] creates and displays a sequential substitution system (SSS) and its causal network, using *ruleset* (as a list of rules, or a RSS index), starting with the state *init* (using string notation), allowing the SSS to evolve for *n* steps. If the initial state string is omitted, SSSInitialState is called to provide a sufficiently complex string. Use the option EarlyReturn to give/deny permission to quit early if the SSS can be identified as dead or repeating.) Use option Mode → Silent to suppress display of the sessie. Any other options given are passed on to SSSDisplay.

(Returns a copy of the SSS that can then be displayed or manipulated without rebuilding, using SSSDisplay, SSSAnimate, or directly, looking at its keys, "Evolution" and "Net", etc.)

Once the sessie has been created you can display it (using SSSDisplay), change options, look at the network, etc., without reconstructing it.

In[ ]:= **? SSSDisplay**

Out[ ]=

Symbol

SSSDisplay[*sss*, *opts*] displays the sequential substitution system *sss*
and/or its causal network. Use SSS (or SSSInitialize and SSSEvolve) to construct it first.

Options:

Min → *n* cuts off the display before the first *n* steps
of the system. (Separate values can be specified for SSSMin and NetMin.)

Max → *n* cuts off the display after the first *n* steps
of the system. (Separate values can be specified for SSSMax and NetMax.)

VertexLabels → Automatic (or "Name") |
"VertexWeight" | … labels vertices by node number or distance from origin, etc.

HighlightMethod → Dot | Frame | Number (or
True) | None (or False) specifies how the matches in the SSS are highlighted.

ShowRule → Bottom | Top | Left | Right | None (or False)
specifies where to place the rulelist icon relative to the SSS visual display (if shown).

Sizes of display components are specified by the options NetSize, SSSSize,
IconSize and ImageSize (which refers to the pane containing the SSS display and icon).

NetMethod → GraphPlot | LayeredGraphPlot | TreePlot | GraphPlot3D | All | NoSSS | list of methods,
where NoSSS generates no SSS display (causal
network only) and the other choices specify how the causal network is to be shown.

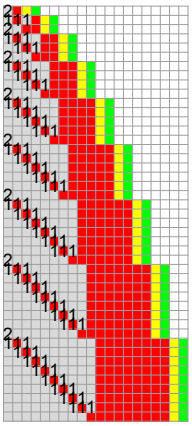
You may choose options interactively using SSSInteractiveDisplay, then click one of the “use” buttons for one time or default use.

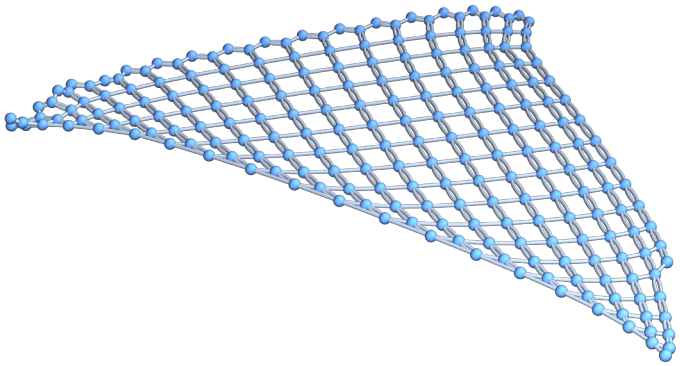
In[ ]:= **SSSInteractiveDisplay[sss2]**

We might chose the GraphPlot3D display for the network. (Clicking and dragging with the mouse shows different view points.)

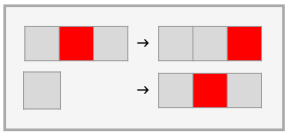
In[ ]:=
```
SSSDisplay[sss2, Min → 1, Max → 500, SSSMin → Automatic, SSSMax → 45, NetMin → Automatic,
 NetMax → 224, HighlightMethod → Number, RulePlacement → Bottom, NetMethod → {GraphPlot3D},
 ImageSize → 170.`, NetSize → 350, SSSSize → {Automatic, 220}, IconSize → {Automatic, 20},
 VertexSize → Automatic, VertexLabels → Placed[Automatic, Tooltip], DirectedEdges → False]
```

Out[ ]=





Substitution Rules:



SSSAnimate shows the order the network is created.

In[•]:=
```
SSSAnimate[sss1]
```

Out[•]=

Other components of the Sessie object can also be studied individually.

In[•]:=
```
sss1[["RuleSet"]]
```

Out[•]=
```
{ABA → AAB, A → ABA}
```

In[•]:=
```
sss1[["Evolution"]]
```

Out[•]=
```
{AB, ABAB, AABB, ABAABB, AABABB, AAABBB, ABAAABBB, AABAABBB, AAABABBB, AAAABBBB,
 ABAAAABBBB, AABAAABBBB, AAABAABBBB, AAAABABBBB, AAAAABBBBB, ABAAAAABBBBB,
 AABAAAABBBBB, AAABAAABBBBB, AAAABAABBBBB, AAAAABABBBBB, AAAAAABBBBBB,
 ABAAAAAABBBBBB, AABAAAAABBBBBB, AAABAAAABBBBBB, AAAABAAABBBBBB, AAAAABAABBBBBB}
```

In[•]:=
```
sss1[["RulesUsed"]]
```

Out[•]=
```
{2, 1, 2, 1, 1, 2, 1, 1, 1, 2, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1}
```

The RuleSet contains the “laws of nature” for this toy universe, rules which are always invoked sequentially: The first rule is always used before the second, if possible, and any rule is always used in the leftmost position possible in the state string. The Evolution gives the list of all state strings from the initial state on. For instance, sss1 only contains the first 25 time steps. The colored box picture is a visual

representation of these state strings, starting with the initial state string at the top, with time flowing downward in the graphic. The component RulesUsed gives the list of which rules were used at each time step.

FAQ Sessie Intro, 2024.11.13, Kenneth Caviness and Colton Edelbach