OVERVIEW OF ORIGINAL EEXIST MECHANICS

So the original system was an extension of the zero-bit computer, wherein every instruction was a simple *transfer* command. The following changes are made:

- instead of being integer addresses, SRC and DST are real-valued, allowing a theoretically continuously-addressed memory;
- instead of a discrete set of instructions, there is a continuum of instructions;
- instead of sequentially executing one instruction after another, all instructions are executed simultaneously;
- instead of a transfer from SRC to DST simply copying data from one location to another, there is a gradual transfer that runs across a period of time, moving more and more of the contents of SRC to DST; and
- instead of a transfer from SRC to DST specifying a single transfer, it specifies a continuum of transfer requests, near the specified SRC->DST transfer.

To accommodate these changes, instructions are coded as a mix of a pair of chemicals (named SRC and DST). The amounts of each chemical indicate the addresses of the transfer. Across a one-dimensional region of space (nominally 0<=x<=40), one imagines a set of tubes of chemicals, each specifying a SRC/DST pair. As the diameter of these tubes shrinks, the number of instructions increases, and the accuracy with which a SRC/DST is treated as an address increases. In the limit, the tubes are infinitesimally thin, so each has an exact address (x coordinate).

Current Modifications

The current transfer mechanism feels a bit off in certain ways. For example, as a tube drains, the SRC and DST will tend towards 0, which means the lower-addressed tubes are likely to change more than say a random tube in the middle of the address space. It also means that instructions stored near 0 can't be used to drain other tubes, since those tubes will eventually affect the instructions near 0.

Of course, some of this is deliberate: the cross-interactions, the interplay between code and data, these are imagined to be important, useful characteristics. But without a clear understanding of these interactions, it's difficult to utilize the EEXIST system. For example, trying to evolve certain behaviors near x=0 is likely futile.

Src and Dst Base Levels

In addition to recording the SRC and DST levels of each tube, there's now a per-tube offset applied to each of these chemical concentrations. This is called the BASE, and there are two base levels per tube: BASE.SRC and BASE.DST. These don't change the actual amounts of chemicals, but rather the interpretation of them. For chemical balances of SRC and DST and a base of BASE.SRC and BASE.DST, the specified transfer is from BASE.SRC+SRC → BASE.DST+DST.

Consequences

It's easy to drain tubes, and by loading BASE values into tube X, any desired transfer can be coded and later activated simply by transferring chemicals out of tube X. This seems to

make it easier to code deliberate system-level behaviors.

Normally, addresses of SRC and DST are absolute: if SRC=5 and DST=10, this indicates a transfer from X=5 to X=10. There has been some exploration of, instead, using *relative* addressing. In the above example, if the SRC and DST values (5 and 10) are at, say, location 20, this would specify a transfer from 25 to 30 (20+5 → 20+10). BASE offsets can be used to change absolute transfers to relative transfers: by setting BASE(x)=x, the above 5->10 transfer would be offset by BASE=20, and become a transfer from 25 to 30.

Alternatively, in a system that uses relative addressing, setting BASE=40-x changes it to direct addressing. For example, in a relative-addressing system, the base at X=20 would be 20, so if SRC and DST=5 and 10, these would be BASE-offset to 25 and 30, which, in a relative system would translate to 45 and 50. Assuming wraparound at the edges, this would signify a transfer from 5 to 10. So adding the ability to specify BASE offsets also allows switching between absolute and relative addressing, whichever system is incorporated at the underlying level.

DIAMETER MODIFICATION

Another new change to the system is a per-tube diameter, which controls flow rate from one tube to another. If two tubes S and D are connected, the amount of chemical transfer is normally based on the timestep dt (and possibly the different in chemical amounts between tubes S and D in some implementations). With diameters, the amount of flow is also multiplied by DIAMETER(S) and DIAMETER(D). While chemical amounts specify the source and destination of a transfer, the DIAMETER allows control over how quickly such transfers occur. It also allows certain tubes to be made immutable: a diameter of 0 means no transfers can be made to or from that tube. Again, the cross-interaction between code and data is generally a desirable feature, but it can be selectively disabled (or perhaps just reduced) using the DIAMETER feature.

Note that the BASE and DIAMETER mechanisms are not mutable via chemical balance. These are fixed, system-level characteristics.

SPECIFIC BEHAVIORS THAT CAN BE CODED

To Be Added Later...

GENETIC APROACH TO SYSTEM DESIGN

While some work has resumed in trying to deliberately engineer specific system behaviors, it's also interesting to look at evolving interesting behaviors using a genetic algorithm (GA). The idea of a GA is to build a random population of systems; see how each one behaves on a given task; pick the top performers; combine their configurations to create a new generation of systems; and repeat this process, trying to evolve a system that performs the given task better across generations.

A first example is to simply take the collective SRC/DST amounts per tube as the genetic

code of an individual. There are a number of options for initializing this system, including:

- completely randomized initial chemical amounts;
- randomized amounts in a randomly-selected set of regions;
- smoothly changing SRC and DST amounts across a region via random deltas;
- randomly varying the difference between SRC and DST, while linearly changing one across a region;

and so on.

Mating options include:

- averaging each chemical between the two parents;
- randomly selecting chemical amounts from one parent or the other, across x;
- randomly selecting chemical amounts from one parent or the other, in chunks of some given size across x (where the chunk sizes can be fixed, randomly-generated, or changing from one mating operation to the next);
- trying to characterize the *nature* of a parent's chemical composition, and then randomly generating a new individual that mimics that nature (for example, a large DST vs SRC in one region; nearer to equal levels in the next region; and so on).

With the addition of BASE and DIAMETER variables, the options for a individual's genome become more complex. While completely random initialization might be interesting, it likely needs a much larger population and a much longer evolution time than is necessarily feasible right now.

As an alternative, one might characterize several different types of system configurations, for example:

- having SRC slowly increase, while having DST increase at a slightly higher rate. This has the effect of causing a migration of chemicals from left to right, beginning with the initial SRC to the initial DST.
- Fixing diameter at 0, to freeze a region of chemical balances.
- setting SRC=DST in tube x, which acts as a NOP but will begin to migrate chemicals from near SRC to near DST should X be modified (depending on how the transfer rate is set: concentration-difference vs. absolute).
- changing SRC and DST randomly but in controlled ways: linear change (with randomized start and end); sinusoidal with randomized frequency, phase and amplitude; purely random; random walk; and so on.

The genome can be comprised of a certain number of segments, each of a random type from the above. The width of each segment can also be randomized. Mating can be done by selecting segments from one parent or the other; by combining segments by averaging characteristics; or by any number of other schemes.