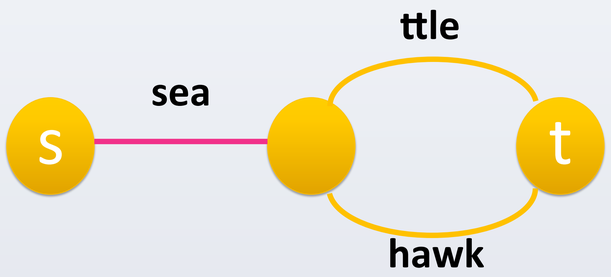
Visualization of Lattice Structure

Shengjie Wang

What is Visualization of Lattice?

Lattice is a special form of the graph, which encodes data on **paths** from the start node to the target node.



Above is a simple lattice example, which encodes data of two words: ***seatlle*** and ***seahawk***.

The most important feature of lattice is the presence of common edges. In the above example, “sea” is the common edge for two paths.

We focus on the common edges for two reasons:

1) the data is compressed as we store the duplicated information only once;

2) it helps to understand the structural information in the data.

In this project, we address the problem of visualizing the lattice structure in an appropriate way, so that users can have easy access to the data encoded as well as the structural information. In particular, we aim to build a system which can *visualize any input lattice* structure, and *support certain editing functionalities*.

Why Visualizing Lattice?

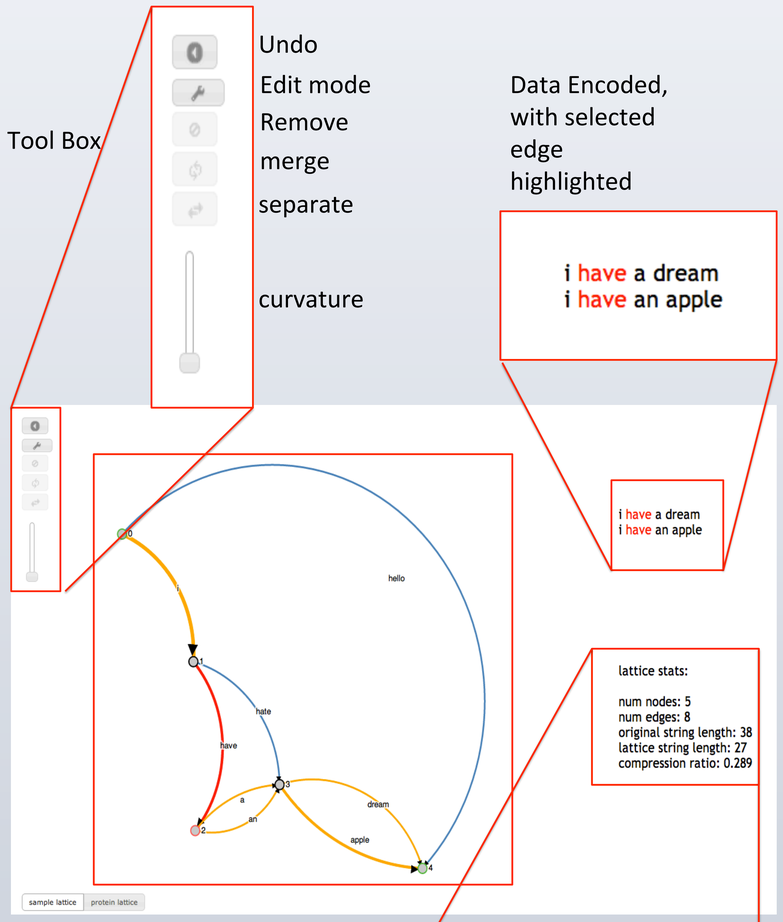
Visualization of lattice structure is an interesting and challenging problem:

1) visualization of graph in general is hard; it has been a research topic for a long time about how to do the layout, and how to display the information;

2) lattice in particular has important features, such as common edges, that have not been addressed by any other graph visualization before;

3) how to construct the lattice structure from raw data is an np-hard problem; by visualizing and interacting with lattice structure, we hope to gain some intuition about how to construct the lattice.

Results



Visualization of lattice, with selected edge and paths highlighted

Lattice Statistics

lattice stats:

num nodes: 5
num edges: 8
original string length: 38
lattice string length: 27
compression ratio: 0.289

Approach

1. Layout of the Lattice Structure:

   Lattice is a special form of graph. We adopt a force-directed layout for lattice, which is a widely-used layout for graph visualization. As lattice always has a start node and a target node, we hold those two nodes fixed. For better exploration, we allow users to set other nodes fixed, as well as adjusting the curvature of edges in the lattice.

2. Display Common Edges and Encoded Data:

   Edges have widths proportional to the number of paths going through. Therefore, common edges are “fatter” and more obvious. Edge lengths are encoded proportional to the data length. User can query data by moving mouse towards any edge, and by using dfs, we return all the data through that edge back to user. User can click on the path, and filter out all the irrelevant edges. Moreover, user can query about certain data, and we return back the relevant paths.

3. Editing Functionalities

   We design to have an editing mode separated from the exploring mode. In the editing mode, we support the following functionalities: add a node to an edge, remove a node, merge two nodes, split a common edge into multiple edges, and merging multiple edges into a common edge.

   With these operations, user can change the lattice to any other possible lattice, and explore how to construct lattice from raw data.

Future Work

Future work may include the following perspectives:

1) get a better layout of the lattice, so that the chance that edges collide will be lower;

2) find some ways to deal with large structures, such as graph clustering;

3) extend to more editing functionalities, and give more flexibilities for user to select nodes and edges.