Fred Morrison
CSC-366

Crypto: Heuristic Problem Solver - Essay 2

A basic explanation of how the program works can be found in task two. As far as some of the additions made in task 4, I've made use of the permutations code from a previous assignment. The permutations code makes it easy to determine whether or not a certain crypto problem fulfills a set of criteria for a heuristic.

For example, if I want to find if two pairs exist in a crypto problem, I can use the permutations code to run through every possible permutation of the problem. If A & B are the same, and C & D are the same, I know there are two pairs.

This is not the most efficient way to do this, however it was very easy to write and also very easy for my mind to grasp. It is for the latter reason that I decided to go with this method.

After making use of the permutations code, most of the seemingly more complex heuristics were extremely easy to write. It made the assignment much less intimidating.

I also made use of the built in module for list functions in SWI-Prolog (:- use_module(library(lists)).). This allowed me to use certain functions that made life easier, such as nth0, which allows you to grab the value of a list at a certain index. I used this method in multiple heuristic implementations.