COMP 597: Assignment #1

Instructor: Xujie Si, TA: Breandan Considine

Due: Oct. 4th before class

This is the first assignment of COMP 597: Automated Reasoning with ML.

1 SAT Encoding

The following solutions can be obtained using a solver of your choice. Please show all work. Only solutions obtained using a SAT solver will receive credit.

Emirpimes (20 points): An *emirp* is a prime whose digits, when reversed, produce a different prime. An *emirpimes* is a semiprime whose reverse is a different semiprime, e.g., 13052869_{10}. What are its factors in base-2? What is the largest emirpimes you can find, whose prime factors are distinct emirps in bases 2 and 10? Please describe your solution and its SAT encoding.

Bonus question (10 points): A *cryptarithm* is a cipher mapping letters to digits, together with a string whose ciphertext satisfies some equation, e.g.:

```
NINETEEN + THIRTEEN + THREE + TWO + TWO + ONE + ONE + ONE = FORTYTWO
42415114 + 56275114 + 56711 + 538 + 538 + 841 + 841 + 841 = 98750538
```

Construct a 20+-character cryptarithm parseable by the following grammar,

$$
\begin{aligned}
&E \to A \mid \ldots \mid Z \mid EE \mid EOE \mid (E) \\
&O \to + \mid \times \mid \div \mid -^{1} \\
&S \to E = E
\end{aligned}
$$

where $\texttt{chars}(E) \neq \texttt{chars}(E')$ and $\texttt{eval}(\texttt{encrypt}(E)) = \texttt{eval}(\texttt{encrypt}(E'))$. Every plaintext word should be defined in the English 10k dictionary.[2] In order to receive credit, it must not be possible to find your cryptarithm (or rewritings thereof) on the internet or in other classmates assignments.

[1] Interpreted in the usual way, but additive and multiplicative identity are forbidden.

[2] https://github.com/first20hours/google-10000-english/blob/master/google-10000-english.txt

2 Problem 2: Build or Improve a SAT Solver

Programming exercise (40 points): Please select one of the following two options, write a short report, and submit your source code. Please provide instructions for how reproduce your findings and a few test cases.

1. Write a SAT solver from scratch by implementing an existing algorithm such as DPLL, unit propagation or two-watched literals, describe your implementation and evaluate it on a few toy SAT problems.

2. Make a substantive improvement to a competitive SAT solver (e.g. Kissat or MiniSat) which measurably increases performance on a standard benchmark, and document your approach and findings.

3 Problem 3: Uninterpreted function equivalence

SMT exercise (40 points): Please typeset a proof sketch using LaTeX, then translate the proof into your favorite SMT solver to construct a specific example or counterexample. Show all work to receive full credit.

1. A polynomial equation whose coefficients and solutions are integers is called *diophantine*. Let $w, x, y, z \in \mathbb{Z}$ and report your solver's largest nontrivial solutions to each of the following diophantine equations: (a) $w = x^3 + y^3 + z^3$ (b) $w^3 + x^3 = y^3 + z^3$ (c) $w^z + x^z = y^z + z$.

2. Prove that $\mathbb{Z}^{n\times n}$ is associative over $\otimes$, and $\otimes$ is distributive over $\oplus$ for some large n. **Bonus** (5 points): Give an example of a nontrivial finite commutative semiring whose elements are matrices and prove it.

3. A nonnegative matrix whose rows and columns all sum to the same number is called *bistochastic*. Find distinct examples $M_1, M_2 : \mathbb{Z}^{n\times n}$ for some large n such that both are nontrivial bistochastic matrices. **Bonus** (5 points): Is $M_i M_j$ is bistochastic for all bistochastic M_i, M_j?

4. Consider the polynomial kernel $\Delta : (\mathbf{f}, \mathbf{g}) \mapsto (\mathbf{f} \cdot \mathbf{g} + r)^q$. The *kernel trick* states $\forall \mathbf{f}, \mathbf{g} : \mathbb{Z}^d$, $\exists \varphi \mid \langle \varphi(\mathbf{f}), \varphi(\mathbf{g}) \rangle = \Delta(\mathbf{f}, \mathbf{g})$. Show the kernel trick holds by finding φ for some large $r, d, q : \mathbb{N}$. What can we say about $\mathcal{O}(\langle \varphi, \varphi' \rangle)$ as $d, q \to \infty$? Is Δ a metric? Prove or disprove it.

5. Prove that 1D discrete convolution, $* : (f, g)[x] \mapsto \sum_{s\in S} f[x - s]g[s]$, over $S = [-j, j]$ for some large value $j \in \mathbb{N}$ is translation equivariant. **Bonus** (10 points): Prove the 2D case for MNIST, i.e., $[0, 255]^{28\times 28}$.

Please submit your answers as a PDF and supplemental work as a ZIP file.