16.410-13 Recitation 3 Problems

Problem 1: Soundness of Arc Consistency Algorithms

Prove that AC-1 algorithm is sound, i.e., the solution returned by the algorithm is indeed an arc consistent network.

Solution The AC-1 algorithm is formalized below.

Algorithm 1: $\texttt{REVISE}(\langle x_i, x_j \rangle)$

```
DELETED ← FALSE;
for each a_i ∈ D_i do
    if there is no a_j ∈ D_j such that ⟨a_i, a_j⟩ ∈ R_ij then
        Delete a_i from D_i;
        DELETED ← TRUE;
return DELETED
```

Algorithm 2: $\texttt{AC} - \texttt{1}$

```
Q ← {⟨x_i, x_j⟩ | ⟨x_i, x_j⟩ ∈ Arcs};
repeat
    CHANGE ← FALSE;
    for each ⟨x_i, x_j⟩ ∈ Q do
        if REVISE(⟨x_i, x_j⟩) = TRUE then
            CHANGE ← TRUE;
until CHANGE = FALSE;
```

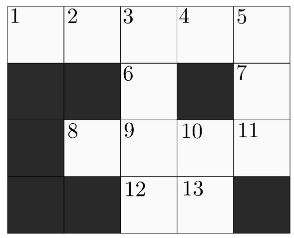
We prove by contradiction. Observe that when the algorithm terminates, the variable `CHANGE` is set to false, which means that the `REVISE` operation returned false for each $\langle x_i, x_j \rangle \in Q$. To arrive at a contradiction, assume that there exists some arc that is not consistent, i.e., there exists some $a_i \in D_i$ for which there is no a_j such that $\langle a_i, a_j \rangle \in R_{i,j}$. But in this case, the `REVISE` operation must have deleted this arc and returned true, when called with $\langle x_i, x_j \rangle$, which contradicts our earlier observation.

Problem 2: Modeling using Constraints

Crossword puzzles can be modeled as constraint satisfaction problems. The task is to assign words from the dictionary into vertical and horizontal slots according to certain constraints. The dictionary is given as follows.

$$\{HOSES, LASER, SHEET, SNAIL, STEER, ALSO, EARN, HIKE,$$
$$IRON, SAME, EAT, LET, RUN, SUN, TEN, YES, BE, IT, NO, US\}$$

The crossword puzzle is provided in the figure below. First, consider a formulation in which each blank



square is a variable. Write down the domains and the constraints. Second, consider a formulation where each starting point for a word (either vertical or horizontal) is a variable. Write the domains and the constraints.

Solution When each blank square is a variable, we have 13 variables, which we denote by $\{x_1, x_2, \ldots, x_{13}\}$. Each variable may take values from the alphabet $\mathcal{A} = \{A, B, C, \ldots, X, Y, Z\}$. Thus, the domain of each variable is $D_i = \mathcal{A}$ for all $i \in \{1, 2, \ldots, 13\}$. Formulating the constraints, in this case, can be carried out by considering the left-to-right and top-to-bottom words in the crossword puzzle. For instance, the variables x_1, x_2, x_3, x_4, and x_5 are constrained so that

$$\langle x_1, x_2, x_3, x_4, x_5 \rangle \in \{\text{HOSES}, \text{LASER}, \text{SHEET}, \text{SNAIL}, \text{STEER}\}$$

Similarly, for instance, the variables x_3, x_6, x_9, x_{12} are constrained as follows

$$\langle x_3, x_6, x_9, x_{12} \rangle \in \{\text{ALSO}, \text{EARN}, \text{HIKE}, \text{IRON}, \text{SAME}\}.$$

In this way, the constraints for variables $\langle x_5, x_7, x_{11} \rangle$, $\langle x_8, x_9, x_{10}, x_{11} \rangle$, $\langle x_5, x_7, x_{11} \rangle$, and $\langle x_{10}, x_{11} \rangle$ can be constructed.

In the second formulation, we have six starting points, namely starting from 1 to left, from 3 to bottom, from 5 to bottom, from 8 to left, from 10 to bottom, and from 12 to left. Let us denote these variables by x_1, x_3, x_5, x_8, x_{10}, and x_{12}. Their domains, denoted by $D_1, D_3, D_5, D_8, D_{10}, D_{12}$, respectively, depend on the number of letters that they are formed by. That is, for five-letter words,

$$D_1 \in \{\text{HOSES}, \text{LASER}, \text{SHEET}, \text{SNAIL}, \text{STEER}\},$$

for four-letter words

$$D_3 = D_8 = \{\text{ALSO}, \text{EARN}, \text{HIKE}, \text{IRON}, \text{SAME}\},$$

for three-letter words,

$$D_5 = \{\text{EAT}, \text{LET}, \text{RUN}, \text{SUN}, \text{TEN}, \text{YES}\},$$

and finally for two-letter words,

$$D_{10} = D_{12} = \{\text{BE}, \text{IT}, \text{NO}, \text{US}\}.$$

The constraints, in this case, depend on the relationship between each pair of variables as they share certain letters. For instance, take variables x_1 and x_3. Clearly, the third letter of x_1 must be the same as the first letter of x_3, which induces the following constraint

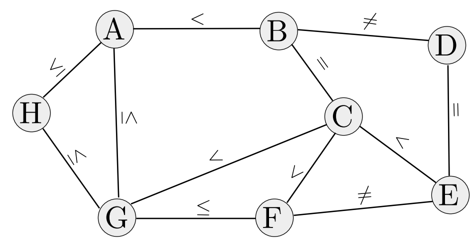
$$\langle x_1, x_3 \rangle \in \{\langle \mathrm{HOSES}, \mathrm{SAME}\rangle, \langle \mathrm{LASER}, \mathrm{SAME}\rangle, \langle \mathrm{SHEET}, \mathrm{EARN}\rangle, \langle \mathrm{SNAIL}, \mathrm{ALSO}\rangle, \langle \mathrm{STEER}, \mathrm{EARN}\rangle\}.$$

Similarly, the constraints between other variables that share certain letters can be extracted in this way.

Notice that in the first formulation we had to use constraints that involve more than two variables, whereas the second formulation involves only binary constraints (constraints with two variables).

Problem 3: Arc consistency

Consider the network given in the figure below. Each node labeled via A, B, C, D, E, F, G, and H take values in $\{1, 2, 3, 4\}$. Find an equivalent arc consistent network (using the AC-1 and AC-3 algorithms).



Solution We provide a partial solution using the AC-1 algorithm and leave the solution using AC-3 algorithm as an exercise. The AC-1 algorithm is described in Algorithms 1 and 2 (see the solution to Problem 1). Roughly speaking, the algorithm tests each arc for *inconsistency* and removes any inconsistent values from the domains of the variables; if a value is removed then the algorithm repeats this procedure until no assignments are removed.

In this case, we have 12 arcs, namely

$$\big\{\langle A,B\rangle, \langle A,G\rangle, \langle A,H\rangle, \langle B,C\rangle, \langle B,D\rangle, \langle C,E\rangle, \langle C,F\rangle, \langle C,G\rangle, \langle D,E\rangle, \langle E,F\rangle, \langle F,G\rangle, \langle G,H\rangle\big\}.$$

The AC-1 algorithm, in its first iteration, goes through every arc to identify inconsistent assignments. That is, first it considers the arc $\langle A,B\rangle$ and runs `REVISE`($\langle A,B\rangle$). This procedure immediately identifies that the value $4 \in D_A = \{1,2,3,4\}$ is indeed inconsistent, i.e., there is no other value $d \in D_B = \{1,2,3,4\}$ such that $4 < d$ is satisfied. Hence, the `REVISE` procedure removes the value 4 from the domain of variable A, and returns `TRUE`. The new domain of variable A is set to $D_A \leftarrow \{1,2,3\}$. The algorithm, then, considers the arc (A,H). Notice tat in this case all variables in the domain of A are indeed consistent. So, no variable is removed. The algorithm examines the remaining arcs in this manner. Since the examination of the first arc by the `REVISE` procedure returned true. The AC-1 algorithm reexamines all the arcs in the next iteration. This is continued until no value is removed from a domain.

You are strongly encouraged to complete this exercise for both the AC-1 algorithm and the AC-3 to better understand how the two algorithms work, and how they differ. Notice that the AC-3 algorithm does the same work using significantly less computation time.