Tom Leighton, Marten van Dijk, and Brooke Cowan

Problem Set 6 Solutions

Due: Tuesday, October 14

Problem 1. [15 points] For each of the following, either prove that it is an equivalence relation and state its equivalence classes, or give an example of why it is not an equivalence relation.

(a) [5 pts] $R = \{(x, y) \in W \times W$ such that the words x and y start with the same letter$\}$, where W is the set of all words in the 2001 edition of the Oxford English dictionary.

Solution. R is an equivalence relation since it is reflexive, symmetric, and transitive. The equivalence class of x with respect to R is the set $[x]_R =$ the set of words y, such that y has the same first letter as x. There are 26 equivalence classes, one for each letter of the English alphabet. ■

(b) [5 pts] $S = \{(x, y) \in W \times W$ such that the words x and y have at least one letter in common$\}$.

Solution. S is reflexive and symmetric, but it is not transitive. Therefore, S is not an equivalence relation. For example, let w_1 be the word "scream," let w_2 be the word "and," and let w_3 be the word "shout." Then $w_3 S w_1$, and $w_1 S w_2$, but it is **not** the case that $w_3 S w_2$. ■

(c) [5 pts] $R := \{(x, y) \in \mathbb{R} \times \mathbb{R}$ such that $\exists n \in \mathbb{Z}, y = 2^n x\}$.

Solution. This is an equivalence relation. We need to show R is reflexive, symmetric, and transitive. First, $x = 2^0 x$, so $(x, x) \in R$, and R is reflexive. Next, if $(x, y) \in R$, then $x = 2^n y$ for some $n \in \mathbb{Z}$, which means that $y = 2^{-n} x$, and so $(y, x) \in R$, and R is symmetric. Finally, if $(x, y) \in R$ and $(y, z) \in R$, then $x = 2^{n_1} y$ and $y = 2^{n_2} z$ for $n_1, n_2 \in \mathbb{Z}$, so $x = 2^{n_1+n_2} z$, and thus $(x, z) \in R$.

The equivalence classes are the infinite sets

$$S_x = \{y \mid \exists n \in \mathbb{Z} \text{ such that } y = 2^n x\}.$$

■

Problem 2. [20 points] In this problem we study partial orders (posets). Recall that a partial order $\preceq$ on a set X is reflexive ($x \preceq x$), anti-symmetric ($x \preceq y \wedge y \preceq x \rightarrow x = y$), and transitive ($x \preceq y \wedge y \preceq z \rightarrow x \preceq z$). Note that it may be the case that neither $x \preceq y$ nor $y \preceq x$. A chain is a list of *distinct* elements $x_1, \ldots, x_i$ in X for which $x_1 \preceq x_2 \preceq \cdots \preceq x_i$. An antichain is a subset S of X such that for all distinct $x, y \in S$, neither $x \preceq y$ nor $y \preceq x$.

The aim of this problem is to show that any sequence of $(n-1)(m-1)+1$ integers either contains a non-decreasing subsequence of length n or a decreasing subsequence of length m. Note that the given sequence may be out of order, so, for instance, it may have the form $1, 5, 3, 2, 4$ if $n = m = 3$. In this case the longest non-decreasing and longest decreasing subsequences have length 3 (for instance, consider $1, 2, 4$ and $5, 3, 2$).

(a) [5 pts] Label the given sequence of $(n-1)(m-1)+1$ integers $a_1, a_2, \ldots, a_{(n-1)(m-1)+1}$. Show the following relation $\preceq$ on $\{1, 2, 3, \ldots, (n-1)(m-1)+1\}$ is a poset: $i \preceq j$ if and only if $i \leq j$ and $a_i \leq a_j$ (as integers).

Solution. We show reflexivity, anti-symmetry, and transitivity. Clearly $i \preceq i$ since $i \leq i$ and $a_i \leq a_i$, so $\preceq$ is reflexive. Next, suppose $i \preceq j$ and $j \preceq i$. Then $i \leq j \leq i$, so $i = j$, and $\preceq$ is anti-symmetric. Finally, suppose $i \preceq j$ and $j \preceq k$. Then $i \leq j$ and $j \leq k$, so $i \leq k$. Moreover, $a_i \leq a_j$ and $a_j \leq a_k$, so $a_i \leq a_k$. Thus, $\preceq$ is transitive. ■

For the next part, we will need to use Dilworth's theorem: Dilworth's theorem states that if $(X, \preceq)$ is any poset whose longest chain has length n, then X can be partitioned into at most n disjoint antichains.

(b) [10 pts] Show that in any sequence of $(n-1)(m-1)+1$ integers, either there is a non-decreasing subsequence of length n or a decreasing subsequence of length m.

Solution. Consider the $\preceq$ relation on $\{1, 2, \ldots, (n-1)(m-1)+1\}$ defined above. The length of the longest non-decreasing subsequence of the given integers is just the length of the longest chain in this poset. If the longest chain has length at least n, we are done, so suppose the length of the longest chain is at most $c \leq n-1$.

Then, by part (a), Dilworth's theorem states that $\{1, 2, \ldots, (n-1)(m-1)+1\}$ can be decomposed into c disjoint antichains. Consider the indices $i_1 \leq i_2 \leq \cdots \leq i_s$ in any antichain A. Then it must be the case that $a_{i_1} > a_{i_2} > \cdots > a_{i_s}$, as otherwise we would have $a_{i_j} \leq a_{i_{j'}}$ for some $j < j'$, and thus $j \preceq j'$, and A could not be an antichain. It follows that there is a decreasing subsequence of length at least $|A|$.

Since we can partition $\{1, 2, \ldots, (n-1)(m-1)+1\}$ into at most $c \leq n-1$ disjoint antichains, one such antichain must have size at least

$$\frac{(n-1)(m-1)+1}{c} \geq \frac{(n-1)(m-1)+1}{n-1} \geq m-1+\frac{1}{n-1} > m-1.$$

So, one antichain must have size at least m, which completes the proof. ■

(c) [5 pts] Construct a sequence of $(n-1)(m-1)$ integers, for arbitrary n and m, that has no non-decreasing subsequence of length n and no decreasing subsequence of length m. Thus in general, the result you obtained in the previous part is best-possible.

Solution. Consider the set of integers $\{1, 2, \ldots, (n-1)(m-1)\}$. For each $1 \leq i \leq n-1$, define the decreasing subsequence of length $m-1$:

$$B_i = i(m-1), \ldots, (i-1)(m-1) + 1.$$

Then the B_i partition $\{1, 2, \ldots, (n-1)(m-1)\}$. Consider the sequence

$$S = B_1 \circ B_2 \circ \cdots \circ B_{n-1}.$$

Any non-decreasing subsequence of S can contain at most one integer from any single B_i, since the B_i are decreasing subsequences. Thus, the length of the longest non-decreasing subsequence is at most $n-1$.

Any decreasing subsequence must be entirely contained in a single B_i, since for $j > i$, any integer in B_j is larger than any integer in B_i. Thus, the length of the longest decreasing subsequence is at most $m-1$. ■

Problem 3. [15 points]

A 3-bit string is a string made up of 3 characters, each a 0 or a 1. Suppose you'd like to write out, in one string, all eight of the 3-bit strings in any convenient order. For example, if you wrote out the 3-bit strings in the usual order starting with 000 001 010..., you could concatenate them together to get a length $3 \cdot 8 = 24$ string that started 000001010....

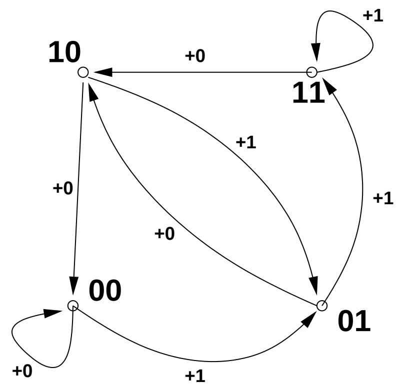
But you can get a shorter string containing all eight 3-bit strings by starting with 00010.... Now 000 is present as bits 1 through 3, 001 is present as bits 2 through 4, 010 is present as bits 3 through 5, ...

(a) [3 pts] Take a few moments to see how short a string you can make that contains every 3-bit string as 3 consecutive bits somewhere in it. Explain why 10 bits is the absolute minimum length for such a string.

Solution. 0001110100 does it with 10 bits and you can't do better: there must be two bits to start and each additional bit can yield at most one new 3-bit string. ■

(b) [3 pts] Imagine that the labels on the vertices of the directed graph below represent the last two digits in a string you build by adding one bit at a time. Explain why the graph completely describes how the last two digits of your string can change throughout this process.

Solution. No matter what the last two bits of your current string x are, say ab, there is a vertex representing that. And, no matter what bit you want to use to extend x, say c, there is an edge directed from your current state, ab, to the vertex bc that represents your new state, the last two bits of xc. Furthermore, these are the only types of vertices and edges in the graph. As a bonus, every edge (s, s') is also labeled with the bit that was added to the string when going from the state s to s'. ■



(c) [3 pts] Find a directed path in this graph starting at some vertex, v, that traverses every edge exactly once. Note that vertices will have to be used more than once and the path wil have to end in v.

Solution. Many: $00, 00, 01, 11, 11, 10, 01, 10, 00$ is one. ■

(d) [3 pts] Explain how such a path provides a shortest possible solution to the original problem.

Solution. Build a solution x using the path. Start with x equal to the two digits in the label of the first vertex on the path. Then for every edge used after that, add on the bit labelling that edge to the string. For example, the path given in last part's solution, yields the string $x = 0001110100$.

Since there are 8 edges, the string will be of length 10, the minimum possible.

Furthermore, the label of the first vertex of the graph, followed by the label of the first edge, give the first 3-bit substring in x. The next vertex and edge, give the next 3-bit substring, and so on. Since all possible 2-bit vertex labels appear exactly once, and each vertex has a 1 edge and a 0 edge departing from it, every 3-bit string corresponds to a unique vertex/out-edge combination. The path uses every possible vertex/out-edge combination exactly once, so the string contains every 3-bit sequence exactly once. ■

(e) [3 pts] What about k-bit substrings, $k = 4, 5, \ldots$? Can you define the appropriate generalization of the useful graph above? (They're called de Bruijn graphs.) If you do it sucessfully, you should be able to see that the in-degree (as well as the out-degree) of every vertex is 2.

Solution. There are 2^k possible k-bit substrings. We can create 2^{k-1} nodes representing the different possible combinations of the first $k-1$ bits. The last bit will be represented as two edges out of each node (0 or 1) and two edges into each node (0 or 1).

It is a theorem that if the in-degree is equal to the out-degree at every vertex of a digraph (and if the graph is connected when all the edges are considered undirected edges) then a directed path can be drawn in that digraph that uses every edge exactly once. You might want to think about why this should be true or how you might find such a path.

But if you do believe it, you should be able to see why all 2^k k-bit strings can be written as substrings of a string of length $2^k + k - 1$. (These strings are essentially de Bruijn strings.) ■

Problem 4. [30 points] Louis Reasoner figures that, wonderful as the Beneš network may be, the butterfly network has a few advantages, namely: fewer switches, smaller diameter, and an easy way to route packets through it. So Louis designs an N-input/output network he modestly calls a *Reasoner-net* with the aim of combining the best features of both the butterfly and Beneš nets:

> The ith input switch in a Reasoner-net connects to two switches, a_i and b_i, and likewise, the jth output switch has two switches, y_j and z_j, connected to it. Then the Reasoner-net has an N-input Beneš network connected using the a_i switches as input switches and the y_j switches as its output switches. The Reasoner-net also has an N-input butterfly net connected using the b_i switches as inputs and the z_j switches as outputs.

In the Reasoner-net the minimum latency routing does not have minimum congestion. The *latency for min-congestion* (LMC) of a net is the best bound on latency achievable using routings that minimize congestion. Likewise, the *congestion for min-latency* (CML) is the best bound on congestion achievable using routings that minimize latency.

Fill in the following chart for the Reasoner-net and briefly explain your answers.

diameter	**switch size(s)**	**# switches**	**congestion**	**LMC**	**CML**

Solution.

diameter	**switch size(s)**	**# switches**	**congestion**	**LMC**	**CML**
$\log N + 4$	2×2	$3N(\log N + 1)$	1	$2 \log N + 3$	$\sqrt{N}$

The diameter of a Reasoner-net is the smaller diameter of the two components plus 2 (to connect to switch to input/output). The diameter of the butterfly component is $\log N + 2$, while the diameter of the Beneš component is $2 \log N + 1$, so overall diameter is 2 + diameter of butterfly $= \log N + 4$.

The number of switches is the number of input and output switches in the Reasoner-net, $2N$, plus the number of switches in its butterfly component, $N(\log N + 1)$, and its Beneš component, $2N \log N$.

The congestion is the congestion of the better of the two component nets, which is the congestion of the Beneš component.

The LMC for the butterfly net equals its diameter, and likewise for the LMC of the Beneš net. So the LMC of the Reasoner-net is 2 plus the LMC of the routing through the component with minimum congestion, namely, 2 plus the diameter of the Beneš net.

The CML equals the congestion of the routing through the component with minimum latency, namely, the congestion of the butterfly net. ■

Problem 5. [20 points] Let B_n denote the butterfly network with $N = 2^n$ inputs and N outputs, as defined in Lecture Notes 10. We will show that the congestion of B_n is exactly $\sqrt{N}$ when n is even.

Hints:

- For the butterfly network, there is a unique path from each input to each output, so the congestion is the maximum number of messages passing through a vertex for any matching of inputs to outputs.
- If v is a vertex at level i of the butterfly network, there is a path from exactly 2^i input vertices to v and a path from v to exactly 2^{n-i} output vertices.
- At which level of the butterfly network must the congestion be worst? What is the congestion at the node whose binary representation is all 0s at that level of the network?

(a) [10 pts] Show that the congestion of B_n is at most $\sqrt{N}$ when n is even.

Solution. First we will show that the congestion is at most $\sqrt{N}$.

Let v be an arbitrary vertex at some level i. Let S_v be the set of inputs that can reach vertex v. Let T_v be the set of outputs that are reachable from vertex v.

By the hint, we have $|S_v| = 2^i$ and $|T_v| = 2^{n-i}$. The number of inputs in S_v that are matched with outputs in T_v is at most $\min\{2^i, 2^{n-i}\}$. To obtain an upper-bound on the congestion of the network, we need to find the maximum value of $\min\{2^i, 2^{n-i}\}$, where the maximum is taken over all i. The maximum value is achieved when 2^i and 2^{n-i} are as equal as possible. Since n is even, these two quantities are equal when $i = n/2$, hence the maximum congestion is

$$2^{n/2} = N^{1/2} = \sqrt{N}.$$

■

(b) [10 pts] Show that the congestion achieves $\sqrt{N}$ somewhere in the network and conclude that the congestion of B_n is exactly $\sqrt{N}$ when n is even.

Solution. We concluded that the congestion of $\sqrt{N}$ can be achieved only at a node at level $\frac{n}{2}$. Consider the node at that level whose binary representation is all 0s. Any packet from the input in the form $z\underbrace{0\ldots000}_{n/2\ bits}$ with destination $\underbrace{000\ldots0}_{n/2\ bits}z'$, where z and z' are any $\frac{n}{2}$-bit numbers, must pass through this node. But there are $2^{n/2} = \sqrt{N}$ of them, giving the node load $\sqrt{N}$. Therefore, we can conclude that the congestion of B_n is exactly $\sqrt{N}$ when n is even. ■