Wqos: homework assignment and midterm exam (v4)

Jean Goubault-Larrecq
LSV, ENS Paris-Saclay

Turn your solution in by October 22nd, 2020!

Note: be as clear as possible. If I cannot understand you, this won’t help you obtain a good grade. Specific hints: (1) write, do not scribble; (2) if I use a specific notation, use the same; (3) use the definitions and results of the lecture notes—if you know an equivalent definition from the literature but the equivalence is not proved in the lecture notes, do not use it (as a last resort, include the proof of equivalence); (4) justify every claim you make, by a proof, by a definition, by a previous question, or by a theorem (preferentially use theorem names, such as “Higman’s Lemma”, rather than numbers); (5) find the simplest possible proof argument; (6) use theorems, not their proofs.

Also, the final exam may contain sequels to some of the questions asked here—who knows.

1 Stuttering compatibility

We fix a transition system \((X, \rightarrow, \leq)\) on a quasi-ordered set \((X, \leq)\).

We recall that \(\rightarrow\) is strongly monotonic if and only if, for all \(x, y, x' \in X\) such that \(x \rightarrow x'\) and \(x \leq y\), there is a \(y' \in X\) such that \(y \rightarrow y'\) and \(x' \leq y'\).

Instead, we say that \(\leq\) is transitively compatible (with \(\rightarrow\)) if and only if for all \(x, y, x' \in X\) such that \(x \rightarrow x'\) and \(x \leq y\), there is a run \(y = y_0 \rightarrow y_1 \rightarrow \cdots \rightarrow y_n\) with \(n \geq 1\) (i.e., the run is non-empty) to some element \(y_n\) such that \(x' \leq y_n\).

It is stuttering compatible if and only if for all \(x, y, x' \in X\) such that \(x \rightarrow x'\) and \(x \leq y\), there is a run \(y = y_0 \rightarrow y_1 \rightarrow \cdots \rightarrow y_n\) with \(n \geq 1\) such that \(x \leq y_1, x \leq y_2, \ldots, x \leq y_{n-1}\), and \(x' \leq y_n\).

For each \(x \in X\), the reachability tree \(FRT(x)\) starting from \(x\) is built as in the final paragraph of the proof of Proposition 1.36 of the lecture notes (page 17), with a slight difference. The root of \(FRT(x)\) is labeled \(x\). The vertices \(v\) of \(FRT(x)\) are in one to one correspondence with the runs \(x = x_0 \rightarrow x_1 \rightarrow \cdots \rightarrow x_k = y\) that are either bad (for all \(i, j\) with \(0 \leq i < j \leq k, x_i \not\leq x_j\)) or minimally good (i.e., not bad, with \(k \geq 1\) and such that \(x = x_0 \rightarrow x_1 \rightarrow \cdots \rightarrow x_{k-1}\) is bad). Such a vertex \(v\) is labeled with \(y\), and we write \(v : y\) for short to say this. The predecessor of a non-root vertex \(v\) corresponding to a run \(x =
\(x_0 \to x_1 \to \cdots \to x_k = y\) \((k \geq 1)\) is the vertex corresponding to the prefix run \(x = x_0 \to x_1 \to \cdots \to x_{k-1}\). A vertex \(w\) is a successor of a vertex \(v\) if and only if \(w\) is not the root and \(v\) is the predecessor of \(w\). A leaf is a vertex with no successor. A live leaf is one whose corresponding run is minimally good. A dead leaf is one whose corresponding run is bad.

One can think of \(FRT(x)\) as being built as follows: we start with just one vertex labeled \(x\), and mark it as unexplored. While there is an unexplored vertex \(v\) corresponding to a run \(x = x_0 \to x_1 \to \cdots \to x_k = y\), either \(y\) is larger than or equal to a vertex \(x_i\) \((0 \leq i < k)\) previously encountered on the branch that leads to \(v\) in the tree, in which case we mark \(v\) as explored and declare it to be a live leaf (then); otherwise, we graft as many unexplored successors \(w : z\) to \(v\) as there are elements \(z\) such that \(y \to z\) as successors of \(y\), and we mark \(y\) as explored (in case \(v\) has no successor, then \(v\) is a dead leaf).

As in the lecture notes, if \(\leq\) is wqo and \((X, \to)\) is image-finite, then \(FRT(x)\) is a finite tree for every \(x \in X\).

**Question 1.** Show that, if \(\leq\) is wqo, and if \(\to\) is transitively compatible, then for every \(x \in X\), there is an infinite run starting from \(x\) if and only if \(FRT(x)\) has at least one live leaf.

**Question 2.** A computation starting from \(x \in X\) is a maximal run \(x = x_0 \to x_1 \to \cdots \to x_k \to \cdots\), namely either an infinite run or a run that ends in a state \(x_k\) such that \(x_k \to x\) for no \(x \in X\). Show that, if \(\leq\) is wqo, if \(\to\) is stuttering compatible, and if \(U\) is any upwards-closed subset of \((X, \leq)\), then for every \(x \in X\), there is a computation starting from \(x\) whose elements are all in \(U\) if and only if \(FRT(x)\) has a path \(v_0 : x_0 \to v_1 : x_1 \to \cdots \to v_k : x_k\) where \(x_0 = x\), \(v_k : x_k\) is a leaf, and \(x_0, x_1, \cdots, x_k\) are all in \(U\).

**Question 3.** The maintainability problem is the following. As input, we are given a state \(x \in X\) and finitely many goal states \(y_1, \ldots, y_n\). We wish to know whether there is a computation \(x = x_0 \to x_1 \to \cdots \to x_k \to \cdots\) starting from \(x\) such that every \(x_i\) is above some \(y_j\) \((i.e.,\, for\, every\, i,\, there\, is\, a\, j \,with\, 1 \leq j \leq n\, such\, that\, y_j \leq x_i)\).

Show that, if \(\leq\) is wqo and decidable, if \(\to\) is stuttering compatible, and if \((X, \to, \leq)\) is image-finite and Post-effective, then the maintainability problem is decidable. Be careful to mention where each assumption is used.

**Question 4.** The inevitability problem is the following. The input is as for the maintainability problem. We wish to know whether all computations starting from \(x\) will eventually go through some state that is above no \(y_j\). Show that, if \(\leq\) is wqo and decidable, if \(\to\) is stuttering compatible, and if \((X, \to, \leq)\) is image-finite and Post-effective, then the inevitability problem is decidable.
2 A wqo on the set of regular languages

We fix a wqo \((X, \leq)\). As usual, \((X^*, \leq_*)\) is the wqo of finite words on \(X\), with the embedding qo. We write \(\epsilon\) for the empty word, \(ww'\) for the concatenation of two words \(w\) and \(w'\). Every letter \(x\) gives rise to a one letter word, again written as \(x\).

Given two subsets \(A\) and \(B\) of \(X^*\), we define:

\[
A.B \overset{\text{def}}{=} \{w w' \mid w \in A, w' \in B\}
\]

\[
A^n \overset{\text{def}}{=} \begin{cases} 
\{\epsilon\} & \text{if } n = 0 \\
A A^{n-1} & \text{if } n \geq 1
\end{cases}
\]

\[
A^* \overset{\text{def}}{=} \bigcup_{n \in \mathbb{N}} A^n
\]

The regular expressions are expressions given by the syntax:

\[
E, F, \ldots ::= \emptyset \quad \text{empty language}
\]

\[
| \epsilon \quad \text{empty word}
\]

\[
| x \quad \text{one-element language } (x \in X)
\]

\[
| E.F \quad \text{concatenation}
\]

\[
| E + F \quad \text{union}
\]

\[
| E^* \quad \text{Kleene star}
\]

The semantics is given by:

\[
\llbracket 0 \rrbracket \overset{\text{def}}{=} \emptyset
\]

\[
\llbracket \epsilon \rrbracket \overset{\text{def}}{=} \{\epsilon\}
\]

\[
\llbracket x \rrbracket \overset{\text{def}}{=} \{x\}
\]

\[
\llbracket E.F \rrbracket \overset{\text{def}}{=} \llbracket E \rrbracket \cdot \llbracket F \rrbracket
\]

\[
\llbracket E + F \rrbracket \overset{\text{def}}{=} \llbracket E \rrbracket \cup \llbracket F \rrbracket
\]

\[
\llbracket E^* \rrbracket \overset{\text{def}}{=} \llbracket E \rrbracket^*
\]

The regular subsets of \(X^*\) are those of the form \(\llbracket E \rrbracket\), for some regular expression \(E\). They form a set \(\mathbb{P}_{\text{reg}}(X^*)\).

You may know that regular languages are exactly the languages recognizable by finite automata... but that is only valid for finite alphabets, and we are considering general, possibly infinite, wqo alphabets \(X\) here. The theory of automata simply does not apply here.

We quasi-order regular subsets by the Hoare quasi-ordering \(\leq^b\). The aim of this part is to show that \(\mathbb{P}_{\text{reg}}(X^*)\) is wqo (assuming \(X\) wqo, naturally).

Equivalently, we quasi-order regular expressions by the following quasi-ordering \(\leq\):

\[
E \leq F \text{ iff } \llbracket E \rrbracket \leq^b \llbracket F \rrbracket,
\]  

(1)
and we will show that $\text{Reg}(X)$, the set of regular expressions on $X$ quasi-ordered by $\preceq$, is wqo (if $X$ is wqo).

**Question 5.** Show that, for any two regular expressions $E$ and $F$:

(a) $E \preceq E + F$ (and, symmetrically, $F \preceq E + F$);
(b) if $\llbracket E \rrbracket \neq \emptyset$, then $E \preceq E.F$ (and, symmetrically, if $\llbracket E \rrbracket \neq \emptyset$, then $F \preceq E.F$);
(c) if $E \preceq E'$ and $F \preceq F'$ then $E + F \preceq E' + F'$;
(d) if $E \preceq E'$ and $F \preceq F'$ then $E.F \preceq E'.F'$;
(e) if $E \preceq E'$ then $E^* \preceq E'^*$;
(f) if $E \preceq F$ then $E \preceq F^*$.

**Question 6.** The normalized regular expressions are: first, the regular expression $0$, and second, the regular expressions defined by the following grammar:

$$E \downarrow, F \downarrow, \ldots := \epsilon$$

$$\mid x \quad \text{for} \quad x \in X$$

$$\mid E \downarrow \cdot F \downarrow$$

$$\mid E \downarrow + F \downarrow$$

$$\mid E \downarrow^*.$$

Show that, for every regular expression $E$, there is a normalized regular expression $E'$ such that $\llbracket E \rrbracket = \llbracket E' \rrbracket$.

Let $\text{Reg}^\downarrow(X)$ denote the set of normalized regular expressions on $X$.

We also define the size $|E|$ of a regular expression $E$ in the obvious way:

$$|0| \stackrel{\text{def}}{=} |\epsilon| \stackrel{\text{def}}{=} |x| \stackrel{\text{def}}{=} 1$$

$$|E \cdot F| \stackrel{\text{def}}{=} |E + F| \stackrel{\text{def}}{=} |E| + |F| \quad |E^*| \stackrel{\text{def}}{=} |E| + 1.$$  

We can then compare regular expressions by size: $E \triangleleft F$ if and only if $|E| < |F|$.

With this, we can order infinite sequences $(E_n)_{n \in \mathbb{N}}$ of regular expressions by the lexicographic extension of $\triangleleft$: $(E_n)_{n \in \mathbb{N}}$ is strictly below $(F_n)_{n \in \mathbb{N}}$ in that ordering if and only if there is an index $n \in \mathbb{N}$ such that $|E_0| = |F_0|, \ |E_1| = |F_1|, \ldots, |E_{n-1}| = |F_{n-1}|$, and $|E_n| < |F_n|$.

**Question 7.** Imagine that there is a bad sequence in $\text{Reg}^\downarrow(X)$, namely a sequence $(E_n)_{n \in \mathbb{N}}$ of normalized regular expressions such that $E_i \preceq E_j$ for no $i < j$. Why is there a minimal bad sequence of normalized regular expressions? By minimal, we mean minimal in the lexicographic extension of $\preceq$.

**Question 8.** Let $(E_n)_{n \in \mathbb{N}}$ be a bad sequence of regular expressions. Show that no $E_n$ is the regular expression $0$ or $\epsilon$. 

4
Question 9. Let again $(E_n)_{n \in \mathbb{N}}$ be a bad sequence of regular expressions. Show that there cannot be infinitely many indices $n$ such that $E_n$ is a one-element language $x_n \in X$. Recall that $X$ is assumed wqo.

Question 10. Let now $(E_n)_{n \in \mathbb{N}}$ be a minimal bad sequence of normalized regular expressions. The previous questions show that no $E_n$ is of the form 0 or ε, and that there is a rank $n_0$ such that for every $n \geq n_0$, $E_n$ is not a one-element language. Show that the set $I$ of indices $n \geq n_0$ such that $E_n$ is a Kleene star $F^*_n$ is finite.

Question 11. Let again $(E_n)_{n \in \mathbb{N}}$ be a minimal bad sequence of normalized regular expressions. Let us imagine that there are infinitely many indices $n \geq n_0$ such that $E_n$ is a union $F_n + F'_n$. Explicitly, let $I$ be an infinite subset of natural numbers larger than or equal to $n_0$ such that, for every $n \in I$, $E_n$ is written as $F_n + F'_n$. Let $\mathcal{F} \overset{\text{def}}{=} \{ F_n | n \in I \}$, $\mathcal{F}' \overset{\text{def}}{=} \{ F'_n | n \in I \}$.

(a) Show that $\mathcal{F}$ is wqo under $\preceq$. ($\mathcal{F}'$ is wqo, too, and the proof is identical.) This is the trickiest question of the whole test.

(b) Why is $\mathcal{F} \times \mathcal{F}'$ wqo?

(c) Derive a contradiction.

Question 12. Let again $(E_n)_{n \in \mathbb{N}}$ be a minimal bad sequence of normalized regular expressions. Mimicking the proof of the previous question, show that there are only finitely many indices $n \geq n_0$ such that $E_n$ is a concatenation $F_n.F'_n$. Please mention the modifications to the previous argument only.

Question 13. Show that $\text{Reg}(X)$ is wqo under $\preceq$.

And now for the coup de Jarnac.

Question 14. Give an alternate proof that $\text{Reg}(X)$ is wqo under $\preceq$ if $X$ is wqo, using Kruskal’s theorem... so that, indeed, what you have done until now directly contradicts recommendation (6) of the introduction (!); but the purposes of the previous questions is to see whether you can do those proofs.