Teoria da Computação

Mini-Teste 1

MIEI 2015/2016
FCT UNL

Consider the cloud dropbox system. The goal of this exercise is to model a simplified version of the system with a structure. The system maintains a global finite collection of folders and a finite collection of clients.

Each folder has a name and a finite collection of files. Each file has a name and a content, the content is represented by an element of set $TEXT$ (do not worry about the definition of $TEXT$).

Each client (user) keeps a name (the userid of the client) and a finite collection of local folders. The local folders are supposed to be local copies of the global folders, but not always, since the local state of a client may be outdated (there are operations that allows users to update their local data from the dropbox and to update the dropbox with their local data, as you will see below).

Let $NAME \stackrel{\text{def}}{=} STRING$ and $USER \stackrel{\text{def}}{=} STRING$.

1. Define the set $FOLDER$ of all folders.
   (a) $FILE \stackrel{\text{def}}{=} NAME \times NAME$ x
   (b) $FILE \stackrel{\text{def}}{=} NAME$ and $FOLDER \stackrel{\text{def}}{=} NAME \times FILE$ x
   (c) $FILE \stackrel{\text{def}}{=} NAME \times TEXT$ and $FOLDER \stackrel{\text{def}}{=} NAME \times \wp(FILE)$
   (d) $FILE \stackrel{\text{def}}{=} NAME \times TEXT$ and $FOLDER \stackrel{\text{def}}{=} NAME \times FILE$ x
   (e) $FILE \stackrel{\text{def}}{=} NAME$ and $FOLDER \stackrel{\text{def}}{=} NAME \times \wp(FILE)$ x
2. Define the set $SDROP$ of all possible dropbox system states.
   (a) $CLIENT \stackrel{\text{def}}{=} USER \times \wp(FOLDER)$ and $SDROP \stackrel{\text{def}}{=} \wp(FOLDER) \times \wp(CLIENT)$
   (b) $CLIENT \stackrel{\text{def}}{=} USER \times FOLDER$ and $SDROP \stackrel{\text{def}}{=} \wp(FOLDER) \times \wp(CLIENT)$ x
   (c) $CLIENT \stackrel{\text{def}}{=} USER \times \wp(FOLDER)$ and $SDROP \stackrel{\text{def}}{=} \wp(FOLDER) \times CLIENT$ x
   (d) $CLIENT \stackrel{\text{def}}{=} \wp(FOLDER)$ and $SDROP \stackrel{\text{def}}{=} \wp(FOLDER) \times \wp(CLIENT)$ x
   (e) $CLIENT \stackrel{\text{def}}{=} USER \times \wp(FOLDER)$ and $SDROP \stackrel{\text{def}}{=} \wp(FOLDER)$ x
3. Define a predicate of First Order Logic that checks if a user u exists in a dropbox system s.
   (a) $\texttt{userExists}(u, s) \stackrel{\text{def}}{=} \forall c.c \in \pi_2(s) \wedge u = \pi_1(c)$ x
   (b) $\texttt{userExists}(u, s) \stackrel{\text{def}}{=} c \in \pi_2(s) \wedge u = \pi_1(c)$ x
   (c) $\texttt{userExists}(u, s) \stackrel{\text{def}}{=} \exists c.c \in \pi_2(s) \rightarrow u = \pi_1(c)$ x
   (d) $\texttt{userExists}(u, s) \stackrel{\text{def}}{=} \exists c.c \in \pi_2(s) \wedge u = \pi_1(c)$
   (e) $\texttt{userExists}(u, s) \stackrel{\text{def}}{=} \exists c.u = \pi_1(c)$ x

4. Define a function that checks if a user exists in a dropbox system.

   (a) $\texttt{hasuser} \subseteq \mathit{SDROP} \to \mathit{BOOL}$
   $\texttt{hasuser} \stackrel{\text{def}}{=} \{(s,u) \mapsto b \mid b = \texttt{userExists}(u,s)\}$ x

   (b) $\texttt{hasuser} \subseteq \mathit{SDROP} \times \mathit{STRING} \to \mathit{BOOL}$
   $\texttt{hasuser} \stackrel{\text{def}}{=} \{(s,u) \mid b = \texttt{userExists}(u,s)\}$ x

   (c) $\texttt{hasuser} \subseteq \mathit{SDROP} \times \mathit{STRING} \to \mathit{BOOL}$
   $\texttt{hasuser} \stackrel{\text{def}}{=} \{(s,u) \mapsto b \mid b = \texttt{userExists}(u,s)\}$

   (d) $\texttt{hasuser} \subseteq \mathit{SDROP} \times \mathit{STRING} \to \mathit{BOOL}$
   $\texttt{hasuser} \stackrel{\text{def}}{=} \{s \mapsto b \mid b = \texttt{userExists}(u,s)\}$ x

   (e) $\texttt{hasuser} \subseteq \mathit{SDROP} \times \mathit{STRING} \to \mathit{NAT}$
   $\texttt{hasuser} \stackrel{\text{def}}{=} \{(s,u) \mapsto b \mid b = \texttt{userExists}(u,s)\}$ x

5. Define a function $\texttt{addclient} \subseteq \mathit{SDROP} \times \mathit{STRING} \to \mathit{SDROP}$ that creates a new client in a dropbox system given its name. When the client is created its name cannot exist yet in the system, so the operation is undefined it it does. The set of local folders is empty after creation.

   (a) $\texttt{addclient} \stackrel{\text{def}}{=} \{(s,u) \mapsto s' \mid s' = (\pi_1(s), \pi_2(s) \cup \{(u,\emptyset)\}) \wedge \neg\texttt{hasUser}(s,u)\}$

   (b) $\texttt{addclient} \stackrel{\text{def}}{=} \{(s,u) \mapsto s' \mid s' = \pi_2(s) \cup \{(u,\emptyset)\} \wedge \neg\texttt{hasUser}(s,u)\}$ x

   (c) $\texttt{addclient} \stackrel{\text{def}}{=} \{(s,u) \mapsto s' \mid s' = (\pi_1(s), \pi_2(s) \subseteq \{(u,\emptyset)\}) \wedge \neg\texttt{hasUser}(s,u)\}$ x

   (d) $\texttt{addclient} \stackrel{\text{def}}{=} \{(s,u) \mapsto s' \mid s' = (\pi_1(s), \pi_2(s) \cup \{(u,\emptyset)\}) \to \neg\texttt{hasUser}(s,u)\}$ x

   (e) $\texttt{addclient} \stackrel{\text{def}}{=} \{(s,u) \mapsto s' \mid s' = (\pi_1(s), \pi_2(s)) \wedge \neg\texttt{hasUser}(s,u)\}$ x

6. Inductively define the set $\mathcal{W}$ of non-empty binary words.

   (a) $0 \in \mathcal{W}$, $1 \in \mathcal{W}$, $w \in \mathcal{W} \to 0w \in \mathcal{W}$, and $w \in \mathcal{W} \to 1w \in \mathcal{W}$

   (b) $0 \in \mathcal{W}$ and $1 \in \mathcal{W}$ x

   (c) $w \in \mathcal{W} \to 0w \in \mathcal{W}$ and $w \in \mathcal{W} \to 1w \in \mathcal{W}$ x

   (d) $0 \in \mathcal{W}$ or $1 \in \mathcal{W}$ or $w \in \mathcal{W} \to 0w \in \mathcal{W}$ or $w \in \mathcal{W} \to 1w \in \mathcal{W}$ x

   (e) $(0 \in \mathcal{W}$ or $1 \in \mathcal{W})$ and $(w \in \mathcal{W} \to 0w \in \mathcal{W}$ or $w \in \mathcal{W} \to 1w \in \mathcal{W})$ x

7. Inductively define the partial function rm on non-empty binary words that removes the rightmost bit from a word of at least lenght 2.

   (a) $rm(w0) = w$ and $rm(w1) = w$ x

   (b) Let $b \in \{0,1\}$. Then, $rm(b0) = b$, $rm(b1) = b$, $rm(w0) = w$, and $rm(w1) = w$

   (c) Let $b \in \{0,1\}$. Then, $rm(b0) = b$, $rm(b1) = b$, $rm(0w) = rm(w)$, and $rm(1w) = rm(w)$ x

   (d) $rm(0) = \varepsilon$, $rm(1) = \varepsilon$, $rm(0w) = rm(w)$, and $rm(1w) = rm(w)$ x

   (e) $rm(0) = \varepsilon$, $rm(1) = \varepsilon$, $rm(0w) = w$, and $rm(1w) = w$ x

8. The set of all finite-length sequences of natural numbers is:

   (a) Uncountable, as the set is the countable union of finite Cartesian products. x

   (b) Countable, as the set is the uncountable union of finite Cartesian products. x

   (c) Countable, as the set is the countable union of infinite sets. x

   (d) Uncountable, as the set is the uncountable union of finite Cartesian products. x

   (e) Countable, as the set is the countable union of finite Cartesian products.