Part A: Modeling with Lattices

**Definition 1** (Partial Order). A partial order is a set of elements $E$ and a relation $\sqsubseteq$ between them s.t.

(a) $\forall x \in E$, $x \sqsubseteq x$ (relation $\sqsubseteq$ is reflexive).

(b) $\forall x, y \in E$, if $x \sqsubseteq y$ and $y \sqsubseteq x$ then $x = y$ (relation $\sqsubseteq$ is anti-symmetric).

(c) $\forall x, y, z \in E$, if $x \sqsubseteq y$ and $y \sqsubseteq z$ then $x \sqsubseteq z$ (relation $\sqsubseteq$ is transitive).

**Definition 2** (Join and meet). Given a partial order $(E, \sqsubseteq)$, and three elements $x, y, z \in E$.

(a) Element $z$ is a least upper bound or join of $x$ and $y$, denoted $x \sqcup y$, if $x \sqsubseteq z$, $y \sqsubseteq z$ (it is an upper bound) and $\forall z' \in E$ with $x \sqsubseteq z'$, $y \sqsubseteq z'$ it is the case that $z \sqsubseteq z'$ (it is the least of the upper bounds).

(b) Element $z$ is a greatest lower bound or meet of $x$ and $y$, denoted $x \sqcap y$, if $z \sqsubseteq x$, $z \sqsubseteq y$ (it is an lower bound) and $\forall z' \in E$ with $z' \sqsubseteq x$, $z' \sqsubseteq y$ it is the case that $z' \sqsubseteq z$ (it is the greatest of the lower bounds).

**Definition 3** (Lattices). A join semi-lattice is a partial order with a least upper bound between every two elements. A meet semi-lattice is a partial order with a greatest lower bound between every two elements. A lattice is a partial order with both.

**Exercise 1.** Let $E \overset{\text{def}}{=} \{1, 2, 3\}$. In each of the cases below, determine whether $(E, \sqsubseteq)$ is a partial order and if it is not, give a reason why.

(a) $\sqsubseteq \overset{\text{def}}{=} \{(1, 2), (2, 3), (1, 3)\}$, that is $1 \sqsubseteq 2$, $2 \sqsubseteq 3$, and $1 \sqsubseteq 3$.

**Answer:** It is not a partial order because reflexivity requires $1 \sqsubseteq 1$, $2 \sqsubseteq 2$, and $3 \sqsubseteq 3$.

(b) $\sqsubseteq \overset{\text{def}}{=} \{(1, 1), (2, 2), (3, 3), (1, 2), (1, 3)\}$.

(c) $\sqsubseteq \overset{\text{def}}{=} \{(1, 1), (2, 2), (3, 3), (1, 2), (2, 3), (1, 3), (3, 1)\}$. 
Exercise 2. Let $\mathcal{E} \overset{\text{def}}{=} \{a, b, c, d\}$. In each of the cases below, determine whether $(\mathcal{E}, \sqsubseteq)$ is a join-semi lattice and if it is not, give a reason why.

(a) $\sqsubseteq \overset{\text{def}}{=} \{(a, a), (b, b), (c, c), (d, d), (a, b), (b, c), (a, c)\}$.

(b) $\sqsubseteq \overset{\text{def}}{=} \{(a, a), (b, b), (c, c), (d, d), (a, c), (a, d), (b, c), (b, d)\}$.

(c) $\sqsubseteq \overset{\text{def}}{=} \{(a, a), (b, b), (c, c), (d, d), (a, c), (a, d), (b, c), (b, d), (c, d)\}$.

Modeling with Lattices  A hospital’s information system incorporates a label tracking system to aid confidentiality and retention policy compliance. The system tags all data with labels indicating its content and/or the encryption (if any) scheme applied to it. It then tracks these labels along with data as they are processed through various sub-systems.

The labels tracked by the system are based on the elements of the lattices $(\mathcal{D}, \sqsubseteq_D)$ and $(\mathcal{E}, \sqsubseteq_E)$ shown in Figure 1. The data lattice deals primarily with patient records which are composed of visiting information and a patient’s profile which includes their billing and health information. The relation $\sqsubseteq_E$ in this lattice may therefore be interpreted as “is part of”. Be careful, however, as this interpretation should not be attributed to the orderings on labels as implemented in the information tracking system you will define in this exercise. The encryption lattice categorizes encryption methods along their strength from no encryption at Plaintext to strong encryption such as AES256 and Twofish256.

The system is governed by confidentiality and retention policy that includes these clauses below.

**Clause A** Staff may access health information.

**Clause B** Visiting information must be retained.

**Clause C** Off-site backups that include profiles must be strongly encrypted.

These were converted into a simplified version of Legalese-like policies with the following grammar:
Given policy $P$ and operation $o$ (an object tracking all the necessary labels), the semantics of the policy, written $[P](o)$ map $o$ to either true or false where the meaning of true is that the operation is allowed and false means that the operation is not allowed. In this scheme, there is no third option; a policy must either allow or deny every operation. The semantics of policy evaluation is given below. The “cond” components of these policies represent predicates over requests to be defined as part of the exercises to follow.

$$
[\text{ALLOW cond}] : o \mapsto \text{true whenever cond}(o)
$$

$$
[\text{DENY cond}] : o \mapsto \text{false whenever cond}(o)
$$

$$
[\text{ALLOW cond EXCEPT deny-policy}] : o \mapsto \text{true whenever cond}(o) \text{ and } [\text{deny-policy}](o)
$$

$$
[\text{DENY cond EXCEPT allow-policy}] : o \mapsto \text{false whenever cond}(o) \text{ and not } [\text{allow-policy}](o)
$$

Given the grammar and semantics of policy evaluation, the hospital defined the following three policies for access, deletion, and backup operations. The “always” below refers to a predicate which is always true.
Your task in this exercise will be to convert the textual descriptions of the conditions to formal predicates.

**Policy A** (Access) is evaluated before every data access by an individual.

- **DENY** always
- **EXCEPT** ALLOW those that definitely have health information

**Policy B** (Delete) is evaluated before any data is deleted from the system.

- **DENY** always
- **EXCEPT** ALLOW those that definitely do not have visiting information

**Policy C** (Backup) is evaluated before data is sent to an off-site backup service.

- **DENY** always
- **EXCEPT** ALLOW those that may have profiles
- **EXCEPT** DENY those that may not be strongly encrypted

Note the use of “may” or “definitely” in the condition texts. Due to the imprecision of the tracking system described later in this section, the enforcement system may not be sure whether an operation is being performed on something that is definitely health information, for example.

**Exercise 3.** In each of the cases below, determine whether that operation would be allowed by the policies above and describe why or why not.

(a) Is a member of the staff allowed to access a screen that may contain health or billing information (like the credit card number of a patient)? (enforcement system is unsure which)

**Answer:** No, they are not allowed. The condition for the exception stipulates that it is certain that health information is present which is not the case in this scenario and thus the default (deny) is the result of the access policy.

(b) Similarly to the previous, what about a screen that definitely contains both health and billing information?

**Answer:** Yes, they are allowed access to this screen. The condition for the exception is satisfied in this scenario; the operation definitely involves health information.

(c) Will the system allow the deletion of a record that includes both visiting and billing information?

(d) Will the system allow off-site backups of profiles that are partially encrypted with strong encryption and partially encrypted with weak encryption?
Formalizing Policies using Lattices  To implement a policy compliance checker, the policy conditions must be evaluated as per the policy semantics. The hospital employs the label tracking system for this purpose. **Assuming every component of the hospital system makes use of at least one of its inputs (or both) to produce its output (though it is unknown which one)**, the label tracking system labels a component’s output using solely the labels of its inputs. This operation will be denoted with □ and will need be defined in the following exercises.

![Diagram of component input and output]

Exercise 4. In each of the cases below, formalize policies by designing a label lattice (based on the two given in Figure 1), operationalize conditions, and define the label combining operation □ so as to comply with the policies:

(a) Define a lattice \((L, \sqsubseteq)\) and combiner □ for labels, and a predicate cond\((l)\) over said labels that is able to determine whether an access definitely has health information.

**Answer:** Let \(L \overset{def}{=} D, \sqsubseteq \overset{def}{=} \sqsubseteq_{D}\), and \(\square \overset{def}{=} \sqcap_D\) where \(\sqcap_D\) refers to the greatest lower bound in the data lattice. Then let cond\((l)\) be satisfied when Health \(\sqsubseteq_L l\).

Using meet to combine labels means that the system will conservatively under-approximate the labels as they are processed throughout the system. Thus if Health \(\sqsubseteq_L l\) it must be the case that the access does contain health information. However, we cannot be sure whether it does or does not contain health information when this condition is not satisfied, like for the element \(\bot\) that would result from a system component processing data labeled Health and Visits.

**Alternate answer:** Reversing a lattice produces a lattice in which meet and join are switched as compared to the original. A reverse version of the answer above can also be given.

(b) Define a lattice \((L, \sqsubseteq_L)\) and combiner □ for labels, and a predicate cond\((l)\) over said labels that is able to determine whether a deletion definitely does not have visiting information.
(c) Define a lattice \((L, \sqsubseteq_L)\) and combiner \(\Box\) for labels, and two predicate \(cond_1(l)\) and \(cond_2(l)\) over said labels that is able to determine whether a backup may have profiles \((cond_1)\) and whether it may not be strongly encrypted \((cond_2)\).

(d) Define the two conditions in (c) without referring to \(\sqsubseteq_D\) or \(\sqsubseteq_E\).