

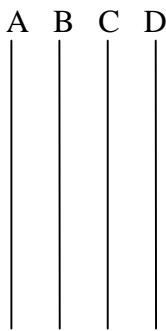
CMSC210 Module B Comprehensive Exercises, 10/14/03

Due at the beginning of the class meeting on 10/17/03

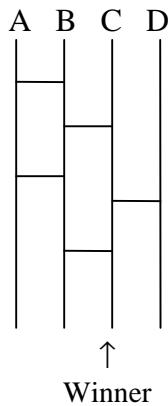
1. “Amida”

In Japan, there is an ancient form of finding the winner among n players. It’s called “amida” (pronounced *ah-mee-dah*, stress each syllable evenly). The process is shown below. First, for n players, prepare n vertical bars, as in (i). Then, each player adds as many horizontal bars as they wish, and the winner is arbitrarily associated with one of the other end of the vertical lines, as in (ii). Note that horizontal bars must be placed so that there is no intersections shaped like ‘+’. Finally, the winner can be found by following the path from the winner position upward to one of the players, moving upward in general except when there is a horizontal path, as in (iii).

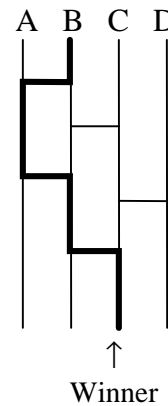
(i) Preparing for 4 players



(ii) Setting up horizontal bars



(iii) Finding the winner



The above example involves only 4 players. But naturally, it can be extended to as many players as we need. Then, it will become more difficult to see the outcome. Behind this simple mechanism, you might see some mathematical properties. For example, how can you be sure that there is exactly one winner? Is there any winning strategy with respect to placing horizontal bars? Is there any relation between the number of bars and how the winner is found? But these are not the questions we discuss in this exercise or in this course. We will explore how to represent amida formally (so that you could use the information for mathematical analysis, programming, etc.).

- A. Formally represent the amida instance as it appears in the figure (iii) above. Call the structure **Amida**, and identify and completely define its components, including “types” where applicable. Make sure that your structure contains the information that is sufficient to recreate the connection pattern and decide the winner as in (iii) (no need to scale). Note that your representation must be extendable to other instances of amida, given different information.

Note: Your structure must contain a set $Players = \{A, B, C, D\}$.

- B. For each relation/function in your structure **Amida** (in Question A above), examine the applicability of all of the following properties **one by one**: surjective, injective, bijective, associative, existence of identity, commutative, reflexive, irreflexive, symmetric, antisymmetric, transitive.

Note: If a property does not apply to a relation/function, write N/A and concisely explain.

2. Crime Scene

Part 1

Not every crime scene is gruesome. You, as a detective, will investigate a potentially only troublesome case described as follows:

Information

1. There is/are footprint(s) on the deck.
 2. If no cats visit the deck, there is no footprints on the deck.
 3. If at least one cat visits the deck, there are more than one cats.
 4. No human have seen cat(s) on the deck.
 5. If a human visits the deck, s/he must see herself/himself on the deck.
 6. There are at least as many humans as cats.
- A. Represent **Information 1** through 5 as statements in First-Order Logic. In addition to the standard symbols in FOL, use only the following symbols specific to the crime scene:
- Unary predicate/relation symbols: *cat*, *human*, *footPrintOnDeck*, *visitDeck*
 - Binary predicate/relation symbol: *seeOnDeck*

Note: Exclude **Information 6** as it involves some advanced formulation.

- B. What can you conclude about the number of cats that visited the deck? Explain by referring to some of the proof techniques discussed in connection to Propositional Logic (ref. Unit B5).

Note: There are multiple crucial steps in this process. Identify all of them.

- C. Suppose that you need to check all the pairs of humans and cats in the crime scene (and you do not even know the entire sets). How would you conclude whether a human saw a cat? Identify the relevant statement in **Information**, and analyze the corresponding FOL statement using some technique for dealing with quantifiers and negation.

Next, consider a structure **CrimeScene** = (*Objects*, *Cats*, *People*, *FootPrintsOnDeck*, *VisitDeck*, *SeeOnDeck*) where

- *Objects* contains all the objects involved in the crime scene.
 - *Cats* and *People* are subsets of *Objects*, and interpret the unary predicate symbols *cat* and *human*, respectively. For example, *human(a)* is true if and only if an object $a \in People$.
 - *FootPrintsOnDeck* defines the meaning of the predicate symbol *footPrintOnDeck*. For example, *footPrintOnDeck(a)* is true if and only if $a \in FootPrintsOnDeck$, i.e., a is a foot print on the deck. Note that $FootPrintsOnDeck \subseteq Objects$.
 - *VisitDeck* defines the meaning of the predicate symbol *visitDeck*, possibly applicable to both Santa Claus and reindeer. For example, *visitDeck(a)* is true if $a \in VisitDeck$, i.e., a visited the deck. Note that $VisitDeck \subseteq Objects$.
 - *SeeOnDeck* defines the meaning of the predicate symbol *seeOnDeck*. For example, *seeOnDeck(a, b)* is true if and only if $(a, b) \in SeeOnDeck$, i.e., a has seen b on the deck.
- D. Do all the cats, if any, need to have visited the deck? Explain.
- E. Could any human have visited the deck? Explain.

- F. Formally define all the structure components of the *smallest* instance of **CrimeScene** that would satisfy all the statements in **Information** shown above. For relations/functions, **give their types** as well. Explain how you came to that conclusion.

Note: The smallest instance would include the minimal number of objects.

Part 2

As a detective, you know that many crime scenes follow similar patterns. So, you conduct a search and found a file on another case with the following information:

Information₂

1. There is/are footprint(s) on the deck.
2. If no youngsters visit the deck, there is no footprints on the deck.
3. If at least one youngster visits the deck, there are more than one youngsters.
4. No human have seen youngster(s) on the deck.
5. If a human visits the deck, s/he must see herself/himself on the deck.
6. There are at least as many humans as youngsters.

For this case, consider a structure **CrimeScene₂** = (*Objects, Youngsters, People, FootPrintsOnDeck, VisitDeck, SeeOnDeck*) where the structure components are defined in a way similar to **CrimeScene**, except the following points:

- Instead of *cat* in **CrimeScene**, another unary predicate/relation symbol *youngster* is used.
- Instead of *Cats*, *Youngsters* is used as a subset of *Objects*, and interprets the unary predicate symbols *youngster*.

- G. Compare **CrimeScene** and **CrimeScene₂** very carefully. What can you conclude?

(Problem 3 on the next page)

3. Ping-Pong

Note: Recall Module A Comprehensive Exercise 3 (Java Comments). In most cases, programming experience was not very helpful. Instead, you needed a good idea about the logic-structure connection. Analogously, you do not need a Ping-Pong experience for this exercise. In case you have little idea about this fascinating ball game, ask your friends. If you really insists that you need to know the complete Ping-Pong rules, visit <http://www.usatt.org/rules/> (but, it is probably useless for this exercise).

- A. Informally describe (in plain English) a single “rally” of Ping-Pong between Players 1 and 2 (i.e., the period during which the ball is in play, beginning with a service until either one fails to return the ball), using a collection of logical statements. Still try to be as precise as possible.

Note: There is a special case where the server fails. In this case, the other player will win.

Hint: Use natural numbers to denote the time when the players serve or return the ball in turn. For example, at Time 0, the server has the ball. If the service is successful, at Time 1, the ball will be with the other player.

- B. Formally define a structure **PingPong** that would satisfy the logical statements in Question A.

Additional instructions:

- Formally define all the structure components including “types” where applicable.
- Formally represent the idea of a “rally” as a function of time (see the hint above), associating with the players.
- Designate one player as the server.
- Identify the winner.
- Make sure that your structure satisfies your logical statements.

Hints/Suggestions:

- As a structure component, include a constant (0-ary function, i.e., returning a fixed value without input) called n without actually specifying the associated value. Use it to indicate the length of the rally. If your structure is further specified with a particular value of n , it could represent the following (and other) situations:
 - $n = 0$: The server failed.
 - $n = 1$: The service was successful, but the other player failed to return the ball.
 - |
 - $n = i$: The rally continued for i service/returns. If i is odd, the server won, otherwise, the other player won.
- Try to make your structure as simple as possible. Do not add structure components unless they are necessary. For example, ignore most properties associated the players, e.g., age, favorite music, and even Ping-Pong experience.

- C. Analyze whether your structure is the only structure that satisfies your logical statements

- D. Would your structure also satisfy the conditions for (lawn) tennis? What does it mean? Explain.

<End>