Overview

Model the operation of a sports complex with multiple fields (soccer, softball, etc.) to schedule games on those fields across different dates without creating any conflicts.

In the sections that follow, names of signatures, relations, facts, predicates, assertions, etc. are in bold red type. You may not add any additional signatures or relations. You may add additional predicates, functions, and assertions, but those described below must exactly match the names and argument list(s) given, and must be defined in a way consistent with the description.

Total Points - 100

Release 1 - Static Model (40 points)

Signatures and Fields (10 points)

There are four signatures: Field, Date, Game, FieldComplex

Exactly one FieldComplex is associated with a schedule of zero or more Game. A Game is associated with a Field (where) and Date (when). Games only appear on the schedule when they have an assigned Field and Date. Games may exist with no assigned field and date; they just do not appear on the schedule.

Facts and Predicates (6 points each, 30 points total)

The following facts in combination define the legal states of the system:

**fact** NotOnSchedule
Games *not* on the schedule have no associated field or date.

**fact** OnSchedule
Games that *are* on the schedule have one field and one date.

**fact** SameField
Games on the same field are scheduled for different dates.

In addition to the facts, the model must have the following run predicates:
pred ScheduledGame
    at least one Game is on the FieldComplex schedule with a Date and Field.

pred UnscheduledGame
    at least one Game is not scheduled.

Use the following run command to explore the state space of the static model:

run{
    ScheduledGame
    UnscheduledGame
} for 5

Release 2 - Dynamic Model (60 points)

Convert the supplied static solution FieldComplex-static.als (found in myCourses->Content->Alloy) into a dynamic model FieldComplex-dynamic.als as described below:

Signatures & Fields (4 points)

Add the ordered Step signature, and make the relations schedule,when,where step-variant.

Facts to Invariant (8 points)

Change all the facts in the static model to predicates with a Step argument. Create a predicate Invariant with a Step argument that is the conjunction of all the facts-turned-predicates.

Initialization (12 points)

init[st : Step]

In the initial state there are no games scheduled. There are consequences for games that must also be explicitly addressed.
Define and run an appropriate init_exists predicate.
Define and check an appropriate init_closed assertion.

Operations (12 points each; 36 points total)

Ten points for each operation predicate per se; three points for the operation's existence predicate (defined and run) and two points for the operation's closure assertion (defined and checked).

scheduleGame[g : Game, f : Field, d : Date, st : Step]

Schedule a new game 'g' in field 'f' at date 'd' if this will not cause a conflict:
- The new game is not currently scheduled
No other games are scheduled on the requested field and date.
The operation completes with the new game scheduled and being specifically
assigned the field and date requested
- Nothing else about the schedule or the other games is affected.

cancelGame[g : Game, st : Step]
Cancel a scheduled game 'g' by removing it from the schedule:
- This may require updating information associated with 'g'.
- The game is currently scheduled
- The operation completes with the game removed from the schedule and releasing the
assigned field and date.
- Nothing else about the schedule or the other games is affected.

rescheduleGame[g : Game, f : Field, st : Step]
Reschedule (if this will not cause a conflict) a currently scheduled game 'g' on its current
date but on field 'f':
- The game is currently scheduled and assigned a field and date
- No other games are assigned the field 'f' on game's 'g' current date.
- The operation completes with the game being assigned the field requested.
- The game's originally assigned date remains the same.
- Nothing else about the schedule or the other games is affected.

Submissions [see course website for due dates]
There are TWO submissions for the project:

- Static Model (R1) - submit only your FieldComplex-static.als model.
- Dynamic Model (R2) - submit both FieldComplex-static.als and FieldComplex-
dynamic.als models. Note that you may optionally elect to use a solution version
of FieldComplex-static.als which will be made available after the R1 due date.

You can submit as many times as you like to the myCourses Dropbox Alloy Project-R1 & Alloy
Project-R2; only the most recent version of each files will be graded.

- The static model must be in a file named exactly FieldComplex-static.als.
- The dynamic model must be in a file named exactly FieldComplex-dynamic.als.
- Do not use zip or any other tool to create an archive - just deposit these two files exactly
as specified.