Refactor

Hui Kwun Tung (53038741)

Building Constraint.java

(before)

|  |
| --- |
| public class BuildingConstraint implements Constraint{ |
| private boolean fulfilled = true; |
|  |
| public BuildingConstraint() {} |
|  |
| public BuildingConstraint(ArrayList<Timeslot> timeslots, ArrayList<String> listOfBldgs) { |
| boolean found = true; |
| for (String i : listOfBldgs) { |
| boolean foundi = false; |
| for (Timeslot s : timeslots) |
| if (i.equals(s.getBuilding())) { |
| foundi = true; |
| } |
| found &= foundi; |
| } |
|  |
| this.fulfilled = found; |
| } |
|  |
| public boolean isFulfilled() { |
| return fulfilled; |
| } |
|  |
| } |

(After)

|  |
| --- |
| public class BuildingConstraint implements Constraint{ |
| private boolean fulfilled = true; |
|  |
| public BuildingConstraint() {} |
|  |
| public BuildingConstraint(ArrayList<Timeslot> timeslots, ArrayList<String> listOfBldgs) { |
| for (String i : listOfBldgs) { |
| for (Timeslot s : timeslots) { |
| if (i.equals(s.getBuilding())) { |
| System.out.println("Found " + i); |
| this.fulfilled = false; |
| break; |
| } |
| } |
| } |
| } |
|  |
| public boolean isFulfilled() { |
| return fulfilled; |
| } |
|  |
| } |

TimeConstraint.java

(Before)

|  |
| --- |
| public class TimeConstraint implements Constraint { |
| private boolean fulfilled = true; |
|  |
| public TimeConstraint(Timeslot t, HashMap<Integer,ArrayList<Double>> daytimeExcluded) { |
|  |
| //daytimeExcluded.put(key, value); |
|  |
| // no course before |
| // no course after |
| // no course between |
| } |
|  |
| private ArrayList<Double> before(ArrayList<Timeslot> list, double t) { |
| double firstTime = 8.0; |
| double lastTime = 22.0; |
| // eg < 10am = 8, 9 |
| // eg >= 3pm = 15, 17, 18, .., 22 |
| return new ArrayList<Double>(); |
| } |
|  |
| private ArrayList<Double> after(ArrayList<Timeslot> list, double t) { |
| double firstTime = 8.0; |
| double lastTime = 22.0; |
| // eg < 10am = 8, 9 |
| // eg >= 3pm = 15, 17, 18, .., 22 |
| return new ArrayList<Double>(); |
| } |
|  |
| private ArrayList<Double> between(ArrayList<Timeslot> list, double t) { |
| double firstTime = 8.0; |
| double lastTime = 22.0; |
| // eg < 10am = 8, 9 |
| // eg >= 3pm = 15, 17, 18, .., 22 |
| return new ArrayList<Double>(); |
| } |
| } |

|  |
| --- |
|  |

(After)

|  |
| --- |
| public class TimeConstraint implements Constraint { |
| private boolean fulfilled = true; |
|  |
| public TimeConstraint(ArrayList<Timeslot> t, HashMap<Integer,ArrayList<Double>> daytimeExcluded) { |
|  |
| for (Timeslot i : t) { |
| if (daytimeExcluded.containsKey(i.getDay())) { |
| for (double j : daytimeExcluded.get(i.getDay())) { |
| if (j < i.getFinishTime() && j >= i.getStartTime()) { |
| this.fulfilled = false; |
| break; |
| } |
| } |
| } |
| } |
|  |
|  |
| } |
|  |
| public boolean isFulfilled() { |
| return fulfilled; |
| } |
|  |
|  |
| } |

Weekday.java

(Before)

Thur(4),

(After)

Thu(4),

Add code

TestSchedule.java

|  |
| --- |
| import schedule.TimeConstraint; import schedule.BuildingConstraint; import schedule.TimeGapConstraint;  // Test case 12: Test time gap constraint |
| @Test |
| public void testTimeGapConstraint() { |
| Timeslot a = new Timeslot("40001","CS3332","C01", "AC1", "LT-1", 14, 16, Weekday.Mon.getDay()); |
| Timeslot b = new Timeslot("40002","CS2332","LA1", "AC2", "5503", 13, 16, Weekday.Tue.getDay()); |
| Timeslot c = new Timeslot("40003","CS3301","LA1", "AC1", "LT-3", 9, 11.5, Weekday.Wed.getDay()); |
| Timeslot d = new Timeslot("40004","CS3201","CA1", "AC3", "6208", 10, 12, Weekday.Tue.getDay()); |
| Timeslot e = new Timeslot("40005","CS3443","CB1", "AC1", "LT-2", 12, 16, Weekday.Tue.getDay()); |
| timeslots.add(a); |
| timeslots.add(b); |
| timeslots.add(c); |
| timeslots.add(d); |
| timeslots.add(e); |
|  |
| TimeGapConstraint rc = new TimeGapConstraint(timeslots, 3); |
| assertEquals(rc.isFulfilled(), true); |
|  |
| Timeslot f = new Timeslot("40006","CS3301","L01", "AC1", "LT-2", 15, 16, Weekday.Wed.getDay()); |
| timeslots.add(f); |
|  |
| rc = new TimeGapConstraint(timeslots, 3); |
| assertEquals(rc.isFulfilled(), false); |
| } |
|  |
| //Test case 13: Test required + time gap constraints |
| @Test |
| public void testRequiredAndTimeGapConstraints() { |
| Timeslot a = new Timeslot("40001","CS3332","C01", "AC1", "LT-1", 14, 16, Weekday.Mon.getDay()); |
| Timeslot b = new Timeslot("40002","CS2332","LA1", "AC2", "5503", 13, 16, Weekday.Tue.getDay()); |
| Timeslot c = new Timeslot("40003","CS3301","LA1", "AC1", "LT-3", 9, 11.5, Weekday.Wed.getDay()); |
| Timeslot d = new Timeslot("40004","CS3201","CA1", "AC3", "6208", 10, 12, Weekday.Tue.getDay()); |
| Timeslot e = new Timeslot("40005","CS3443","CB1", "AC1", "LT-2", 12, 16, Weekday.Tue.getDay()); |
| timeslots.add(a); |
| timeslots.add(b); |
| timeslots.add(c); |
| timeslots.add(d); |
| timeslots.add(e); |
|  |
| ArrayList<String> listOfCrns = new ArrayList<String>(); |
| listOfCrns.add("40001"); |
| listOfCrns.add("40005"); |
| RequiredConstraint rc = new RequiredConstraint(timeslots, listOfCrns); |
| TimeGapConstraint rc1 = new TimeGapConstraint(timeslots, 3); |
| assertEquals(rc.isFulfilled() && rc1.isFulfilled(), true); |
|  |
| listOfCrns.clear(); |
| listOfCrns.add("40001"); |
| listOfCrns.add("40007"); |
| rc = new RequiredConstraint(timeslots, listOfCrns); |
| assertEquals(rc.isFulfilled() && rc1.isFulfilled(), false); |
|  |
| listOfCrns.clear(); |
| listOfCrns.add("40003"); |
| listOfCrns.add("40004"); |
| rc = new RequiredConstraint(timeslots, listOfCrns); |
| Timeslot f = new Timeslot("40006","CS3301","L01", "AC1", "LT-2", 15, 16, Weekday.Wed.getDay()); |
| timeslots.add(f); |
| rc1 = new TimeGapConstraint(timeslots, 3); |
| assertEquals(rc.isFulfilled() && rc1.isFulfilled(), false); |
| } |
|  |
| //Test case 14: Test time constraint |
| @Test |
| public void testTimeConstraint() { |
| Timeslot a = new Timeslot("40001","CS3332","C01", "AC1", "LT-1", 13, 16, Weekday.Mon.getDay()); |
| Timeslot b = new Timeslot("40002","CS2332","LA1", "AC2", "5503", 14, 16, Weekday.Tue.getDay()); |
| Timeslot c = new Timeslot("40003","CS3301","LA1", "AC1", "LT-3", 9, 11.5, Weekday.Wed.getDay()); |
| timeslots.add(a); |
| timeslots.add(b); |
| timeslots.add(c); |
|  |
| HashMap<Integer,ArrayList<Double>> daytimeExcluded = new HashMap<Integer,ArrayList<Double>>(); |
|  |
| ArrayList<Double> l1 = new ArrayList<Double>(); |
| l1.add(9.0); |
| l1.add(10.0); |
| l1.add(12.0); |
| l1.add(18.0); |
| l1.add(19.0); |
| l1.add(20.0); |
| l1.add(21.0); |
| l1.add(22.0); |
| daytimeExcluded.put(1, l1); |
|  |
| TimeConstraint rc = new TimeConstraint(timeslots, daytimeExcluded); |
| assertEquals(rc.isFulfilled(), true); |
|  |
| Timeslot d = new Timeslot("40004","CS3305","LA1", "AC1", "LT-3", 16, 18, Weekday.Mon.getDay()); |
| timeslots.add(d); |
| rc = new TimeConstraint(timeslots, daytimeExcluded); |
| assertEquals(rc.isFulfilled(), true); |
|  |
| Timeslot e = new Timeslot("40005","CS3305","LA1", "AC1", "LT-3", 12, 14, Weekday.Tue.getDay()); |
| timeslots.add(e); |
| rc = new TimeConstraint(timeslots, daytimeExcluded); |
| assertEquals(rc.isFulfilled(), true); |
|  |
| Timeslot f = new Timeslot("40006","CS3305","LA1", "AC1", "LT-3", 12, 14, Weekday.Mon.getDay()); |
| timeslots.add(f); |
| rc = new TimeConstraint(timeslots, daytimeExcluded); |
| assertEquals(rc.isFulfilled(), false); |
|  |
| ArrayList<Double> l2 = new ArrayList<Double>(); |
| l2.add(9.0); |
| l2.add(16.0); |
| l2.add(17.0); |
| l2.add(18.0); |
| l2.add(19.0); |
| l2.add(20.0); |
| l2.add(21.0); |
| l2.add(22.0); |
| daytimeExcluded.put(2, l2); |
| timeslots.remove(f); |
| rc = new TimeConstraint(timeslots, daytimeExcluded); |
| assertEquals(rc.isFulfilled(), true); |
|  |
| Timeslot g = new Timeslot("40007","CS3301","LA1", "AC1", "LT-3", 9, 11, Weekday.Tue.getDay()); |
| timeslots.add(g); |
| rc = new TimeConstraint(timeslots, daytimeExcluded); |
| assertEquals(rc.isFulfilled(), false); |
| } |
|  |
| //Test case 15: Test time constraint and helper functions |
| @Test |
| public void testTimeConstraintWithHelperFunctions() { |
| Timeslot a = new Timeslot("40001","CS3332","C01", "AC1", "LT-1", 14, 16, Weekday.Mon.getDay()); |
| Timeslot b = new Timeslot("40002","CS2332","LA1", "AC2", "5503", 13, 16, Weekday.Tue.getDay()); |
| Timeslot c = new Timeslot("40003","CS3301","LA1", "AC1", "LT-3", 9, 12, Weekday.Wed.getDay()); |
| Timeslot d = new Timeslot("40004","CS3201","CA1", "AC3", "6208", 10, 12, Weekday.Tue.getDay()); |
| Timeslot e = new Timeslot("40005","CS3443","CB1", "AC1", "LT-2", 12, 13, Weekday.Tue.getDay()); |
| timeslots.add(a); |
| timeslots.add(b); |
| timeslots.add(c); |
| timeslots.add(d); |
| timeslots.add(e); |
|  |
| HashMap<Integer,ArrayList<Double>> daytimeExcluded = new HashMap<Integer,ArrayList<Double>>(); |
|  |
| // Mon: Exclude before 12, after 18 |
| ArrayList<Double> mon = new ArrayList<Double>(); |
| mon.addAll(Schedule.beforeTime(12)); |
| mon.addAll(Schedule.afterTime(18)); |
| daytimeExcluded.put(1, mon); |
|  |
| //Tue: Exclude before 9, after 18 |
| ArrayList<Double> tue = new ArrayList<Double>(); |
| tue.addAll(Schedule.beforeTime(9)); |
| tue.addAll(Schedule.afterTime(18)); |
| daytimeExcluded.put(2, tue); |
|  |
| TimeConstraint rc1 = new TimeConstraint(timeslots, daytimeExcluded); |
| assertEquals(rc1.isFulfilled(), true); |
|  |
| //Wed: Exclude between 12 to 14 |
| ArrayList<Double> wed = new ArrayList<Double>(); |
| wed.addAll(Schedule.betweenTime(12, 14)); |
| daytimeExcluded.put(3, wed); |
|  |
| rc1 = new TimeConstraint(timeslots, daytimeExcluded); |
| assertEquals(rc1.isFulfilled(), true); |
|  |
|  |
|  |
| //Wed: Exclude before 13 |
| wed = new ArrayList<Double>(); |
| wed.addAll(Schedule.beforeTime(13)); |
| daytimeExcluded.put(3, wed); |
|  |
| rc1 = new TimeConstraint(timeslots, daytimeExcluded); |
| assertEquals(rc1.isFulfilled(), false); |
|  |
| wed = new ArrayList<Double>(); |
| wed.addAll(Schedule.beforeTime(9)); |
| wed.addAll(Schedule.beforeTime(8)); |
| wed.addAll(Schedule.beforeTime(7)); |
| daytimeExcluded.put(3, wed); |
|  |
| rc1 = new TimeConstraint(timeslots, daytimeExcluded); |
| assertEquals(rc1.isFulfilled(), true); |
| } |
|  |
| // Test case 16: Test building constraint |
| @Test |
| public void testBuildingConstraint() { |
| Timeslot a = new Timeslot("40001","CS3332","C01", "AC1", "LT-1", 14, 16, Weekday.Mon.getDay()); |
| Timeslot b = new Timeslot("40002","CS2332","LA1", "AC2", "5503", 13, 16, Weekday.Tue.getDay()); |
| Timeslot c = new Timeslot("40003","CS3301","LA1", "AC1", "LT-3", 9, 11.5, Weekday.Wed.getDay()); |
| Timeslot d = new Timeslot("40004","CS3201","CA1", "AC3", "6208", 10, 12, Weekday.Tue.getDay()); |
| Timeslot e = new Timeslot("40005","CS3443","CB1", "AC1", "LT-2", 12, 16, Weekday.Tue.getDay()); |
| timeslots.add(a); |
| timeslots.add(b); |
| timeslots.add(c); |
| timeslots.add(d); |
| timeslots.add(e); |
|  |
| ArrayList<String> listOfBuildings = new ArrayList<String>(); |
| listOfBuildings.add("CMC"); |
|  |
| BuildingConstraint rc = new BuildingConstraint(timeslots, listOfBuildings); |
| System.out.println(rc.isFulfilled()); |
| assertEquals(rc.isFulfilled(), true); |
|  |
| listOfBuildings.add("AC3"); |
| rc = new BuildingConstraint(timeslots, listOfBuildings); |
| System.out.println(rc.isFulfilled()); |
| assertEquals(rc.isFulfilled(), false); |
| } |
|  |
| } |