

APPENDIX B

Using SELFPLANNER through its API—An Example

Let us define three activities, two located at the university, and one at the town center. One will be interruptible, one non-interruptible and the final one will be non-interruptible periodic. Completing the non-interruptible one before the interruptible one is a strong preference, but not a necessity.

Listing B.1: Using The SELFPLANNER API Example in Java

```
1 // Initialize the data structures
2 Console console = new Console() {
3     @Override
4     public void write(String s) {
5         System.out.println("SelfPlanner:_" + s);
6     }
7 };
8 int id = 100;
9 Date now = new Date();
10 long ts = now.getTime();
11 TaskManager tmgr = new TaskManager("" + id + "_" + ts);
12 LocationManager lmgr = new LocationManager("" + id + "_" + ts);
13 BinaryManager ordconstraints = new BinaryManager("" + id + "_" + ts, tmgr);
14 TemplateManager templates = new TemplateManager("" + id + "_" + ts);
15 DomainManager dmg = new DomainManager("" + id + "_" + ts);
16 LocationClassManager lcmgr = new LocationClassManager("" + id + "_" + ts);
17 BinaryManager dminconstraints = new BinaryManager("" + id + "_" + ts, tmgr);
18 BinaryManager dmaxconstraints = new BinaryManager("" + id + "_" + ts, tmgr);
19 BinaryManager iconstraints = new BinaryManager("" + id + "_" + ts, tmgr);
20 MyPlannerData data = new MyPlannerData("" + id + "_" + ts, tmgr, lmgr, ordconstraints,
    templates, dmg, lcmgr, dminconstraints, dmaxconstraints, iconstraints);
21 // Create the clear domain template
22 Template clearDomain = new Template(DomainAction.ActionAffects.DAY, "Clear_Domain", -1);
23 clearDomain.setMultipleActions(0, DomainAction.Action.NOT_INC, 48);
24 // Define the problem instance
25 // Locations
```

```

26 Location univ = new Location(0, "University");
27 univ.setLatLng("40.62554905578553_22.960052490234375");
28 lmgr.addLocation(univ);
29 Location center = new Location(1,"Town_Center");
30 center.setLatLng("40.626330777594895_22.9489803314209");
31 lmgr.addLocation(center);
32 // Get the real distance using the Google Distance Matrix API
33 LocationUtils.createLocationPairUsingDistanceMatrix(univ, center, lmgr);
34 try {
35     Thread.sleep(30);
36 } catch (InterruptedException e) {}
37 LocationUtils.createLocationPairUsingDistanceMatrix(center, univ, lmgr);
38 // Enable Traveling Time Minimization
39 data.setLocationDistanceUtility(LOCATION_DISTANCE_UTILITY);
40 // Interruptible Activity
41 Domain d1 = new Domain(startDate, endDate);
42 d1.addTemplate(clearDomain, null, null, DomainAction.Action.NOT_INC);
43 for (Date[] interval : intervalsOfActivity1) {
44     d1.addManualAction(new ManualAction(interval[0],
45         interval[1], DomainAction.Action.INC));
46 }
47 DomainPrefs prefs1 = new NewDomainPrefs(NewDomainPrefs.NLPref.MORNING, TEMPORAL_UNARY_UTIL);
48 Task t1 = new Task(0, "Work_on_project", 150, 300, extraDurationUtil1, "University", d1,
49     prefs1, null, activityUtility1, 1.0, 30, 90, 180, 1440);
49 tmgr.addTask(t1);
50 // Non-Interruptible Activity
51 Domain d2 = new Domain(startDate, endDate);
52 d2.addTemplate(clearDomain, null, null, DomainAction.Action.NOT_INC);
53 for (Date[] interval : intervalsOfActivity2) {
54     d2.addManualAction(new ManualAction(interval[0],
55         interval[1], DomainAction.Action.INC));
56 }
57 DomainPrefs prefs2 = new NewDomainPrefs(NewDomainPrefs.NLPref.AFTERNOON, TEMPORAL_UNARY_UTIL);
58 Task t2 = new Task(1, "Visit_library", 120, 180, extraDurationUtil2, "Town_Center", d1,
59     prefs2, null, activityUtility2, 1.0);
59 tmgr.addTask(t2);
60 // Periodic Activity
61 Domain d3 = new Domain(startDate, endDate);
62 d3.addTemplate(clearDomain, null, null, DomainAction.Action.NOT_INC);
63 for (Date[] interval : intervalsOfActivity3) {
64     d3.addManualAction(new ManualAction(interval[0],
65         interval[1], DomainAction.Action.INC));
66 }
67 DomainPrefs prefs3 = new NewDomainPrefs(NewDomainPrefs.NLPref.AFTERNOON, TEMPORAL_UNARY_UTIL);
68 PeriodicPrefs.Period period = PeriodicPrefs.Period.DAILY;
69 PeriodicPrefs pprefs = new PeriodicPrefs(period, PeriodicPrefs.Range.END_BY_DEADLINE, true,
70     true);
71 Task t3 = new Task(2, "Have_lunch", 30, 60, extraDurationUtil1, "University", d1, prefs3,
72     pprefs, activityUtility3, 1.0);
73 tmgr.addTask(t3);
74 BinaryPreference p1 = new BinaryPreference(1, 0, binaryUtil, BinaryPreference.ORD);
75 ordconstraints.addPreference(p1);

```

```

74 // Call SelfPlanner to solve the problem instance
75 SelfPlannerClient client = new SelfPlannerClient("selfplanner.dai.uom.gr", console,
    PRIV_KEY, data, "-k_7000");
76 Solution[] solutions = null;
77 TaskSolved[] solved;
78 try {
79     solutions = client.simpleConnectAndSolve(NUM_OF_PLANS, univ); // The current location is
    at the university
80 } catch (TooManyClientsException ex) {
81     System.out.println("SelfPlanner_overload_Exiting.");
82     System.exit(1);
83 }
84 if (solutions == null || solutions.length == 0)
85     System.out.println("No_solution_was_returned");
86 else { // Print Plans
87     int pn, pn2;
88     for (int i=0; i<NUM_OF_PLANS; i++) {
89         solved = solutions[i].tasks();
90         pn = i + 1;
91         System.out.println("Printing_plan_" + pn);
92         for (TaskSolved element : solved) {
93             for (int j=0; j<element.solution().length; j++) {
94                 for (int k=0; k<element.solution()[j].length; k++) {
95                     pn = j + 1;
96                     pn2 = k + 1;
97                     System.out.println(element.task().name() + ":Period_" + pn + "," + ",_
    part_" + pn2);
98                     System.out.println(element.solution()[j][k].getStartTime() + ":_:" +
    element.solution()[j][k].getEndTime());
99                 }
100             }
101         }
102     }
103 }

```