READ ME for Project Part 2 Source Code

# Source code

**Programming language**: C# (CSharp)

**.Net Framework:** .NetCore 3.1 (Compatible to Windows and Linux platforms)

**Project source code location:** \SourceCodeFiles\code\scheduler\src

**Test Project Source code location:** \SourceCodeFiles\code\schedulertest

# How to run?

Navigate to the folder \SourceCodeFiles\code\scheduler\bin\Debug\netcoreapp3.1 on command prompt

And run “ai-scheduler.exe” with the following example parameters:

ai-scheduler.exe {self\_country\_name} {resource\_data\_file\_name} {initial\_world\_state\_file\_name} {schedule\_output\_file\_name} {number\_of\_schedules\_to\_print} {the\_search\_depth} {gamma\_val} {c\_val\_failure\_cost} {k\_val\_logistic\_function}

**C:\Users\satishra\Dropbox\MyDocsLib\MS-Vanderbilt\0-Courses\cs-5260-Artificial-Intelligence\Project-Part2-Deliverables\SourceCodeFiles\code\schedulertest\bin\Debug\netcoreapp3.1>**ai-scheduler.exe "Carpania" "C:\Users\satishra\Dropbox\MyDocsLib\MS-Vanderbilt\0-Courses\cs-5260-Artificial-Intelligence\ai-scheduler\initial\_data\resource\_seeding\_data.csv" "C:\Users\satishra\Dropbox\MyDocsLib\MS-Vanderbilt\0-Courses\cs-5260-Artificial-Intelligence\ai-scheduler\initial\_data\country\_resource\_seeding\_data.csv" "C:\Users\satishra\Dropbox\MyDocsLib\MS-Vanderbilt\0-Courses\cs-5260-Artificial-Intelligence\ai-scheduler\output\_data\schedules.txt" 15 7 9, .8, 1

# The schedule output file location

A file named “schedules.txt” will be generated at the {schedule\_output\_file\_name} location. In the example above it’s C:\Users\satishra\Dropbox\MyDocsLib\MS-Vanderbilt\0-Courses\cs-5260-Artificial-Intelligence\ai-scheduler\output\_data\schedules.txt

**Note:** Increasing {the\_search\_depth} parameter above 12 can significantly impact the schedule generation output

# Important source code file

**Program.cs:** Entrypoint to the project containing the program’s main function

**GameScheduler.cs:** The program file containing the main interface function:

/// <summary>

/// Creates the schedule and saves the schedules for my country

/// </summary>

/// <param name="myCountryName">The name of the country that's labeled as my country</param>

/// <param name="resourcesFileName">The csv file name containing names, weights and other information related to the resources</param>

/// <param name="initialWorldStateFileName">The csv file name containing initial state of the world, the resources and it's quantity etc.</param>

/// <param name="outputScheduleFileName">The file name where the output schedule will be saved</param>

/// <param name="numOutputSchedules">The number of ordered list of output schedule to be written to the output schedule file</param>

/// <param name="depthBound">The maximum depth to search assuming the initial depth is 0</param>

/// <param name="gammaValue">Configurable gamma value for the Logistics function</param>

/// <param name="c\_val\_failure\_cost">Configurable C value failure cost factor to compute the expected utility</param>

/// <param name="k\_val\_logistics\_function">Configurable K value to compute the expected utility</param>

public void CreateMyCountrySchedule(

string myCountryName,

string resourcesFileName,

string initialWorldStateFileName,

string outputScheduleFileName,

uint numOutputSchedules,

uint depthBound,

double gammaValue = .9,

double c\_val\_failure\_cost = .8,

double k\_val\_logistics\_function = 1)

**GameManager.cs:** The program file containing the function to evaluate a proposed world state schedule, and the code to generate schedule and random forest

**DataProvider.cs:** The program that reads the resource and initial world state file and converts them into VirtualResource and VirtualWorld objects

**TemplateProvider.cs:** The program with the code to return transfer and transform templates with the yield

**PriorityQueue.cs**: Priority queue for implementing the beam search with the frontier max size, and keeps item (virtual world state) sorted based upon the Expected Utility (EU) of the self

**VirtualWorld.cs:** The code representing virtual world with the code to deep clone the VirtualWorld object to generate successors, a hashtable of applied actions, a pointer to the parent world state

**UtilityCalculator.cs:** Program file containing code for the StateQuality function, discounted and undiscounted reward function, probability for a given country to accept a schedule, the probabilities of the participating countries to accept the schedule, expected utility calculator function

**ScheduleSerializer.cs:** Code file with the logic to generate the schedule in the following format in the schedules.txt output file:

[

(TRANSFER Carpania Atlantis ((Timber 50)) EU:0.6137382786100645

(TRANSFER Carpania Atlantis ((Timber 50)) EU:0.6250475186971308

(TRANSFER Carpania Atlantis ((Timber 50)) EU:0.6341315564342566

(TRANSFER Carpania Atlantis ((Timber 50)) EU:0.6412359737536151

(TRANSFORM Carpania (INPUTS (Population 25) (MetallicElements 5) (Timber 25) (MetallicAlloys 15)) (OUTPUTS (Housing 5) (HousingWaste 5) (Population 25)) EU(self):0.6299843927067952

(TRANSFER Carpania Atlantis ((Timber 50)) EU:0.6353533548534973

(TRANSFORM Carpania (INPUTS (Population 25) (MetallicElements 5) (Timber 25) (MetallicAlloys 15)) (OUTPUTS (Housing 5) (HousingWaste 5) (Population 25)) EU(self):0.6263081998290951

]