Design and Implementation of a NetLogo Interface for the Stand-Alone FYPA System

Daniela Briola
Viviana Mascardhi

WOA 2011 @ UNICAL
The Dynamic Resource Allocation Problem from Ansaldo STS

- Resources are the “platforms (nodes) inside a station”:
  - indivisible
  - non-sharable
- Resources may become unavailable
  - they break up and become available again (when fixed)
- Organized in a directed graph of dependencies (because connected by railway tracks), represented as arcs $R_1 \rightarrow R$, $R_2 \rightarrow R$, ..., $R_n \rightarrow R$ in the graph
- A set of couples of conflicting arcs in the graph of dependencies (C$\rightarrow$D, F$\rightarrow$G)
- Continuous and linear time model
The Dynamic Resource Allocation Problem

- A set of entities (trains) with
  - different priorities
  - needing to use some of the available resources for a predefined time in a predefined time interval
  - a static allocation plan that selects resources in predefined time slots
- They enter and exit the system in any moment
- They can change the chosen resources inside the graph, but never the first and the last resource of the plan
- They can change the resources time usage (duration of usage and interval of usage)
The model of the problem

- A directed and non-planar graph that entities must traverse from one start point to one end point
- “Start points” and “End points” resources (A and H)
- A set of couples of conflicting arcs in the graph of dependencies (C->D, F->G)
  - The usage of C->D during TS makes C-> and F->G occupied for TS
- Continuous and linear time model
The FYPA negotiation protocol

- Able to simulate/manage the real world
  - Trains may use nodes for longer than planned
  - Nodes can break up
- The protocol looks for a dynamic re-allocation of the resources to the entities such that the re-allocation:
  - is free of conflicts
  - is completed within a pre-defined amount of time
  - minimizes as much as possible (sub-optimal solution):
    - the changes between the new plan and the static allocation
    - the delay of the entities in reaching the end point
    - the number of entities and resources involved in the re-allocation process
The Ansaldo STS FYPA system
The Ansaldo STS FYPA system

- Ansaldo STS uses a system able to calculate, twice a year, the global distribution of trains in the various stations
  - However, this system does not calculate the trains allocation inside the stations
- This allocation was made by the human operators
- FYPA system first of all fills this gap: the output of the already existing Ansaldo System became the input of the FYPA system, in the offline version
  - The output of FYPA off line version is the complete allocation of trains to be used in the station
- Then, FYPA can be used online to reallocate trains after delays or node failures
The Ansaldo STS FYPA system (2)

- The FYPA MAS is connected to the Ansaldo STS system using Web Services. FYPA reads inputs and sends outputs using web services.
- The FYPA system output is shown at run time thanks to the Ansaldo STS GUI.
The StandAlone FYPA system

- We separated the FYPA MAS from the Ansaldo STS system to get a stand alone system
- The core of the stand alone version is the same of the Ansaldo STS FYPA
  - We only changed the startup phase and the agents managing the input/output
  - Web services have been substituted by a database
  - A new offline interface has been created
The StandAlone FYPA system: performances

- The performances are the same for both the systems
- Simple examples: within 2 seconds every kind of conflict is solved
- Complex examples executed by Ansaldo STS on real data
  - Every 2 seconds a new train is created
  - Mestre station: 528 Treno agents in a day, 59 Nodo agents, each Nodo agent manages approximately three entering arcs. The total number of incompatibilities is 430. The simulation took 18 minutes to be completed.
  - Pisa station: 395 trains in a day, 60 Nodo agents, each having about 20 entering arcs. Every arc has 130 incompatibilities on average. The simulation took 13 minutes to be completed
Pisa station
Pisa station
The NetLogo interface
NetLogo (1)

- Is a programmable modeling environment for simulating natural and social phenomena
- NetLogo is very useful to simulate the evolution of a system consisting of thousands of simple agents divided into different categories
- In that kind of applications, decisions about what action is to be done are usually made using a probabilistic choice
- Simulated time does not require a sophisticated management
  - the built-in representation of time provided by NetLogo by means of Ticks (a discrete representation of time shared among agents) is enough.
NetLogo (2)

- Its graphical interface contains just a 2D spatial view of the model environment, which is a square lattice.
  - It is very simple to draw on this lattice, using the NetLogo primitives
  - The graphical representation of agents is built-in in NetLogo
- It supports three types of agents: turtles, patches, and links.
  - Patches represent fixed square (in 2D) or box (in 3D) cells on the main 2D (or 3D) view of the world
  - Turtles are agents that can move around on the world surface
  - Links represent relationships between turtles
Why not using NetLogo for FYPA?

- We could not use NetLogo to implement FYPA algorithm mainly because of the following reasons:
  - is not suitable to simulate a negotiation among agents based on a “deterministic view”
  - is not suitable to simulate an interaction where agents exchange messages
  - does not support a continuous model of time
- The existence of multiple railway tracks among two nodes was difficult to model: only one link between each couple of agents is allowed
- The “ask” command is sequential: it’s really difficult to simulate the concurrent choice made by different agents
- We should have forced NetLogo to behave in a completely different way with respect to its own philosophy
The NetLogo FYPA interface

- NetLogo offers an integrated graphical representation of the simulation, so it is almost simple to let the user see how the simulation is going on.
- Our NetLogo program reads the structure of the station and the movements of a set of Trains during the time from the log files that are output by StandaFYPA.
  - Actually manually reworked.
- Nodes and trains are NetLogo turtles, arcs are NetLogo Link agents.
- Time is represented using NetLogo Ticks.
- At every tick, each train will read from a private list (filled with its final path in the graph) and will move on a new node, if necessary.
The NetLogo FYPA interface
Screen shots from NetLogo interface
Future Work

- Improve FYPA system to avoid modifying manually the input files for the NetLogo interface
- Try to make the NetLogo interface an on line interface, automatically updated during the FYPA execution
Thank you!!

Any questions?