Pietro Terna

terna@econ.unito.it

Department of Economic science and Public finance “G.Prato”

University of Torino – Italy

Agents ruling and ruled, rules and meta-rules: does the world belong to anarchists?

web.econ.unito.it/terna

The outline

We propose a framework useful to investigate three topics:

- how enterprises and organizations arise, behave, and fall;
- how agents operate, as ruling or ruled entities;
- how modify rules, to improve them and obtain better results (and how to define results).

The tool is a large agent-based simulation framework reproducing the economic context in a detailed way.

The tool

AESOP (Agents and Emergencies for Simulating Organizations in Python) uses actions, agents (acting and deciding people) and the scheduling of events into an agent based framework.

Agents may use fixed rules but they can also learn to improve these rules in a changing environment. They can also be modeled so as to be aware of the consequences of their behavior.

In the simulator, single agents can also be modeled as neural networks, so that the system appears as a "net of neural networks".

The simulator is currently based on Swarm (www.swarm.org) as basic layer, see http://web.econ.unito.it/terna/jes

The AESOP simulator is under implementation in Python (www.python.org), using **SLAPP, Swarm-Like Agent Protocol in Python**, http://eco83.econ.unito.it/terna/slapp/

[we want to join the easiness of Python, and its openness, to the clarity of the Swarm protocol; Python is connected in AESOP to the R statistical system (R is at http://cran.r-project.org/), via the rpy library, at http://rpy.sourceforge.net/)]

The SWARM protocol and SLAPP

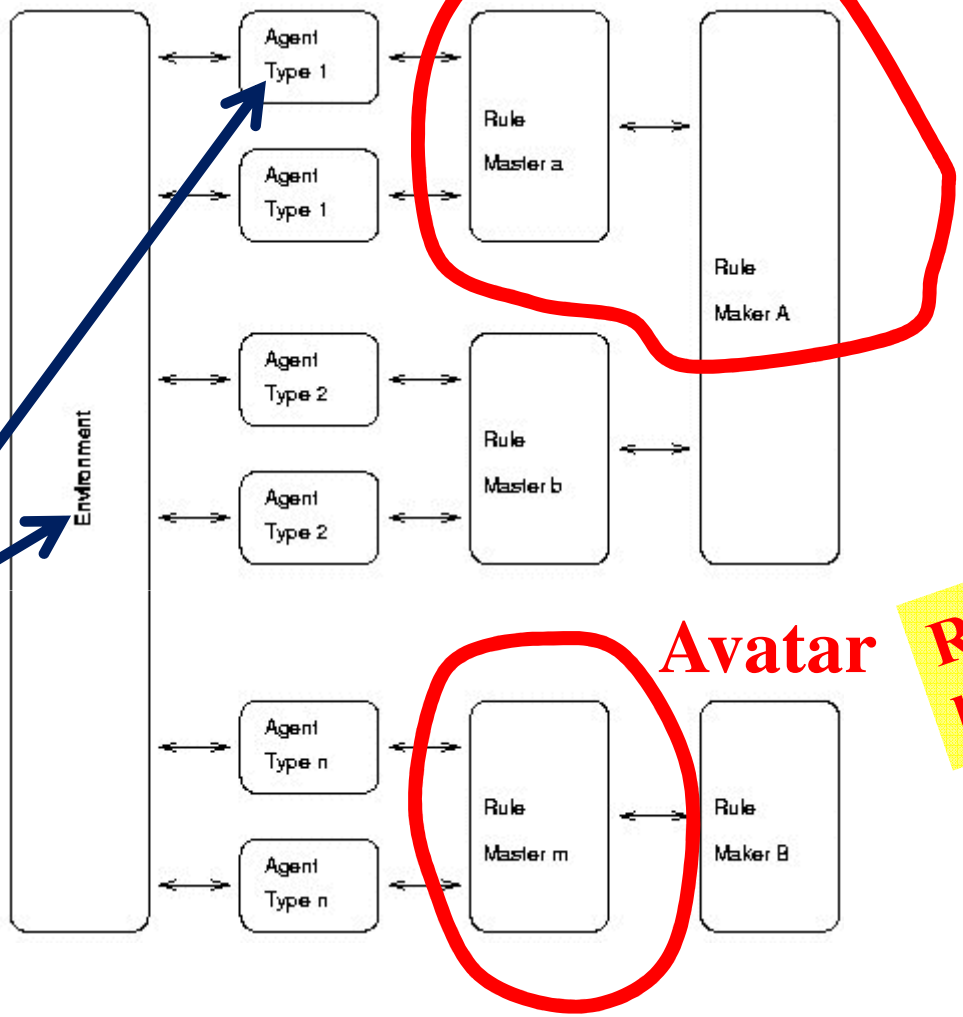
An **absolutely clear and rigorous** application of the SWARM protocol is contained in the original SimpleBug tutorial (1996?) with ObjectiveC code and text by the Swarm development team in Santa Fe (N. Minar, R. Burkhart, C. Langton and M. Askenazi), on line at
http://ftp.swarm.org/pub/swarm/apps/objc/sdg/swarmapps-objc-2.2-3.tar.gz
(into the folder "tutorial", with the texts reported into the README files in the tutorial folder and in the internal subfolders)

The same tutorial has also been adapted to Java by Charles J. Staelin (*jSIMPLEBUG, a Swarm tutorial for Java*, 2000), at
http://www.cse.nd.edu/courses/cse498j/www/Resources/jsimplebug11.pdf (text) or
http://eco83.econ.unito.it/swarm/materiale/jtutorial/JavaTutorial.zip (text and code)

At http://eco83.econ.unito.it/terna/slapp you can find the same structure of files, now implementing the SWARM protocol using Python

So, the SWARM **protocol** as ***lingua franca*** in agent based simulation models

An abstract scheme: Environment, Rules, Agents



Fixed rules

NN

CS

GA

Reinforcement learning

Avatar

Microstructures, mainly related to time and parallelism

http://web.econ.unito.it/terna/ct-era/ct-era.html

Existing code (a Java layer upon a JavaSwarm implementation, on line at http://web.econ.unito.it/terna/jes/

WD - DW - WDW simplified formalism

Existing code (a Java layer upon a JavaSwarm implementation, on line at http://web.econ.unito.it/terna/jes/

- **WD** side or formalism**: What to Do**

- **DW** side or formalism**: which is Doing What**

- **WDW** formalism: **When Doing What**

WD, What to Do

recipe = a sequence of steps to be executed

DW, which is Doing What

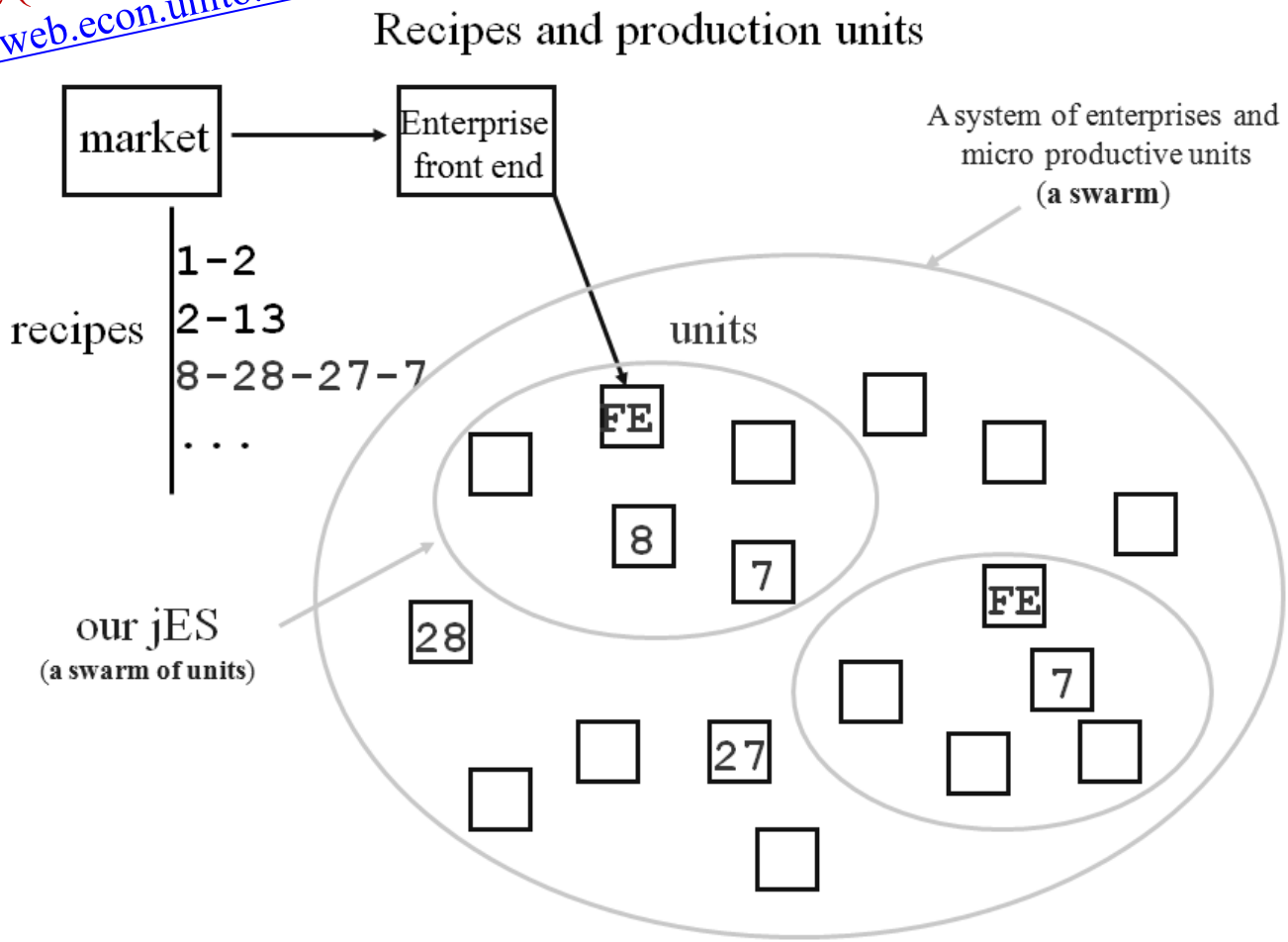
unit = a productive structure within an organization or operating alone; a unit is able to perform one (or more) of the steps described in a recipe

WDW, When Doing What

a clock

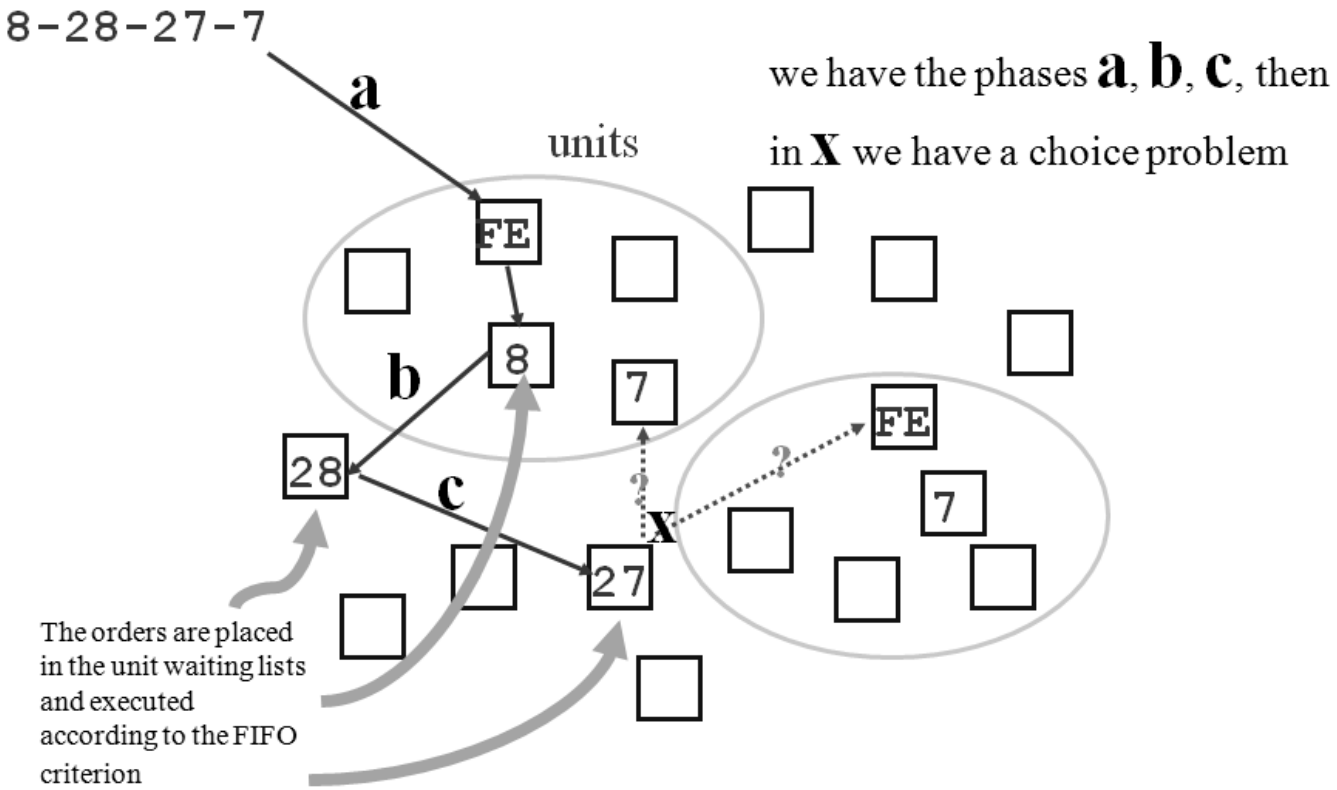
Existing code (a Java layer upon a JavaSwarm implementation, on line at http://web.econ.unito.it/terna/jes/

Existing code (a Java layer upon a JavaSwarm implementation, on line at http://web.econ.unito.it/terna/jes/



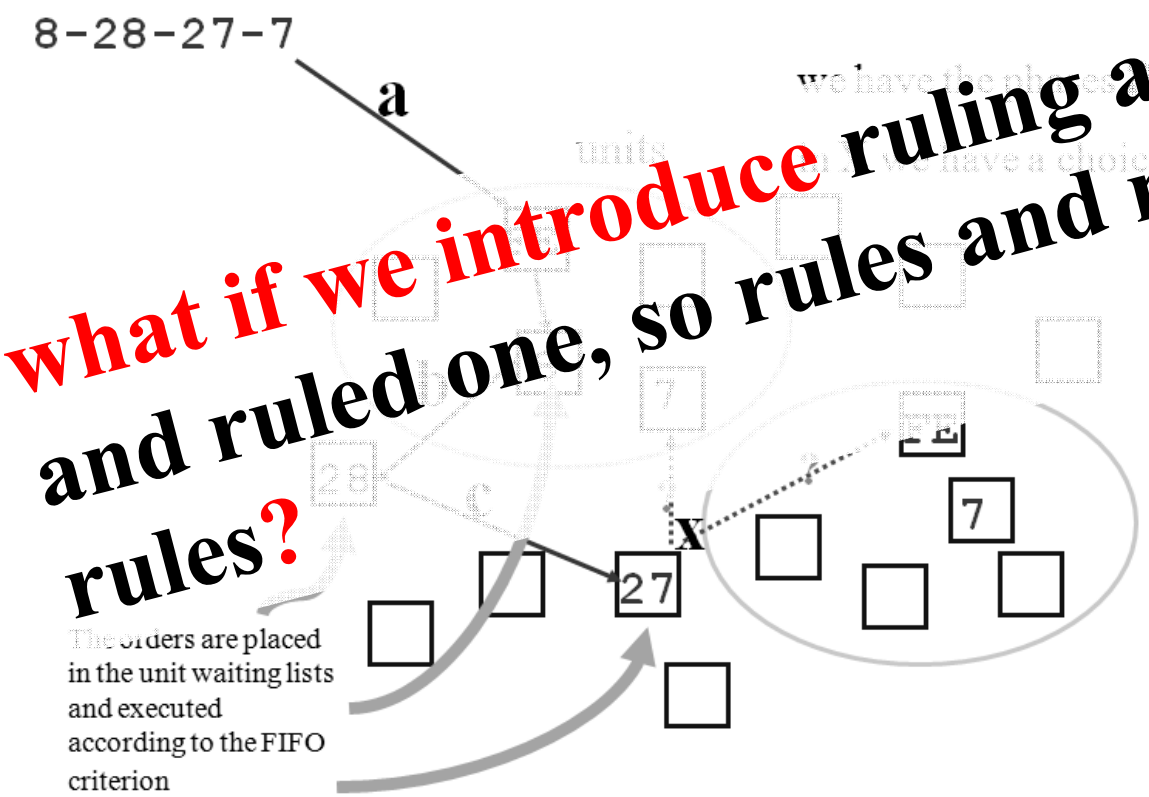
Existing code (a Java layer upon a JavaSwarm implementation, on line at http://web.econ.unito.it/terna/jes/

Recipes on move



Existing code (a Java layer upon a JavaSwarm implementation, on line at http://web.econ.unito.it/terna/jes/

Recipes on move



what if we introduce ruling agents and ruled one, so rules and meta-rules?

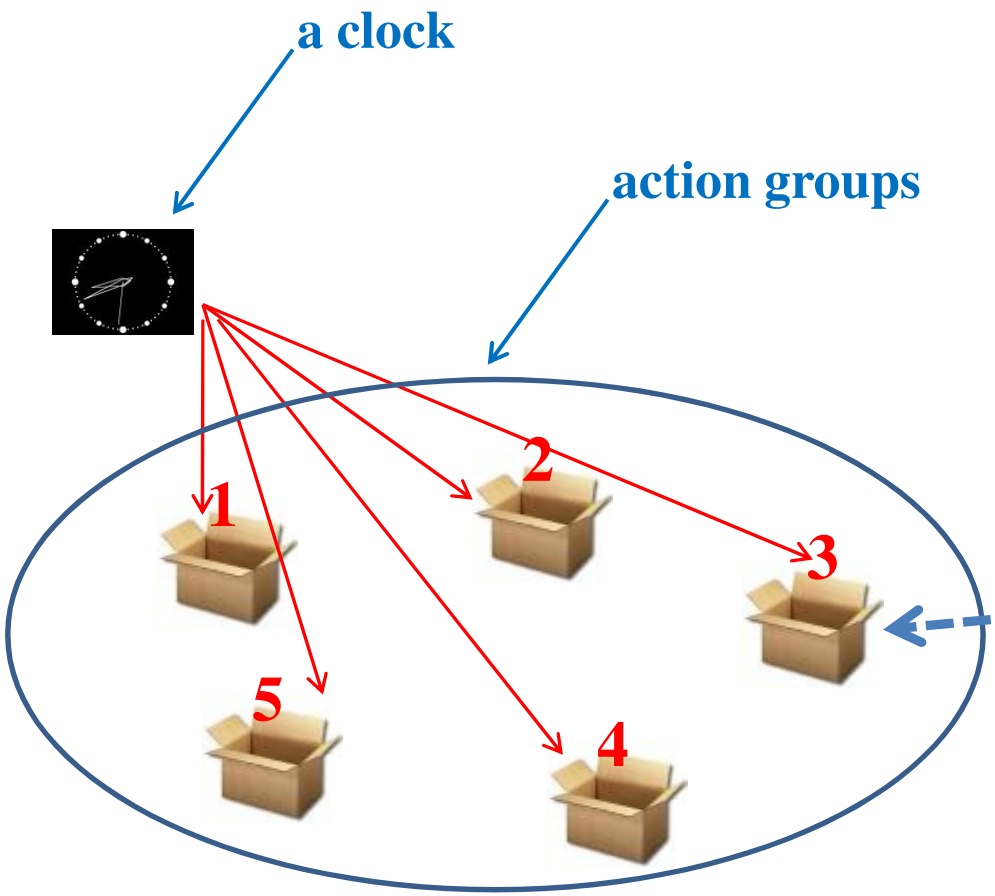
The existing jES (Java Enterprise Simulator, with applications) tool is introduced in E. Mollona (ed.), Computational analysis of firms' organization and strategic behavior, Routledge, forthcoming (2009)

* * *

The main idea for the evolution of the simulator framework is that of introducing:

- rule modifications (**adaptive** WD side of the model) and
- agent adaption to changing rule (**adaptive** DW side of the model)

A new schedule approach



What in each box?

Tasks to be executed (with p=1 or with p<1)

Tasks are included into the code in a static way, or can be added/activated dynamically by other tasks, also via agents' actions

Tasks can be read – via a 'read' task schedule element – from an external source (file, web interaction, …)

A special type of task to be read from an external source is that of the recipes

tasks read from an external archive

a_n – a specific agent (instance of class A)
a_X – a randomly chosen agent (instance of class A)
a_all – all the agents (instances) of the class A

methods specific of each agent or inherited from the basic type ‘agent’

recipes read from an external archive

[[agent method] [agent method] [agent method] [agent method] [agent method] …]

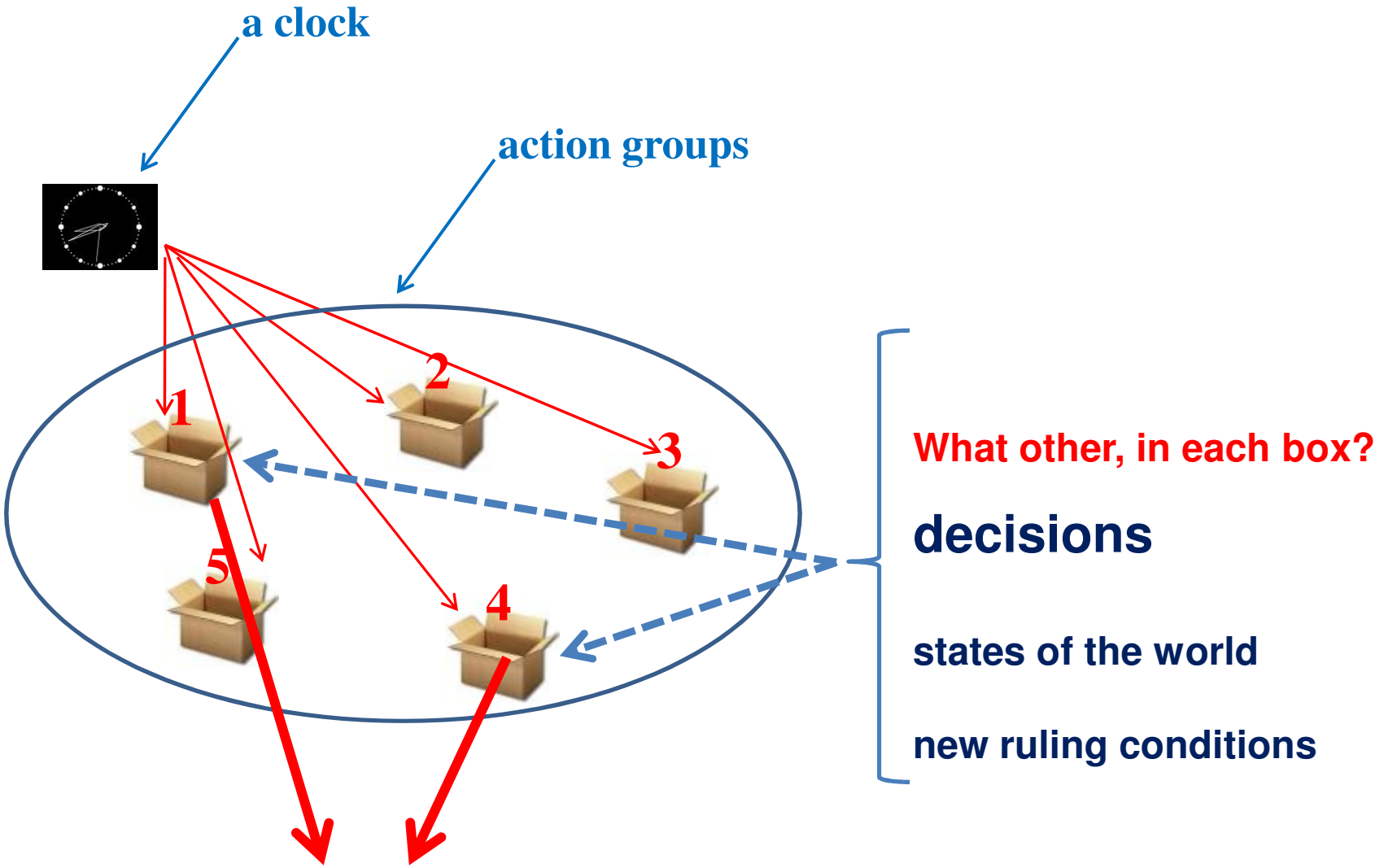
to be executed in a sequential way by a given thread of agents (static or dynamic)

[[agent method] [[agent method] [agent method] [agent method]] [agent method] …]

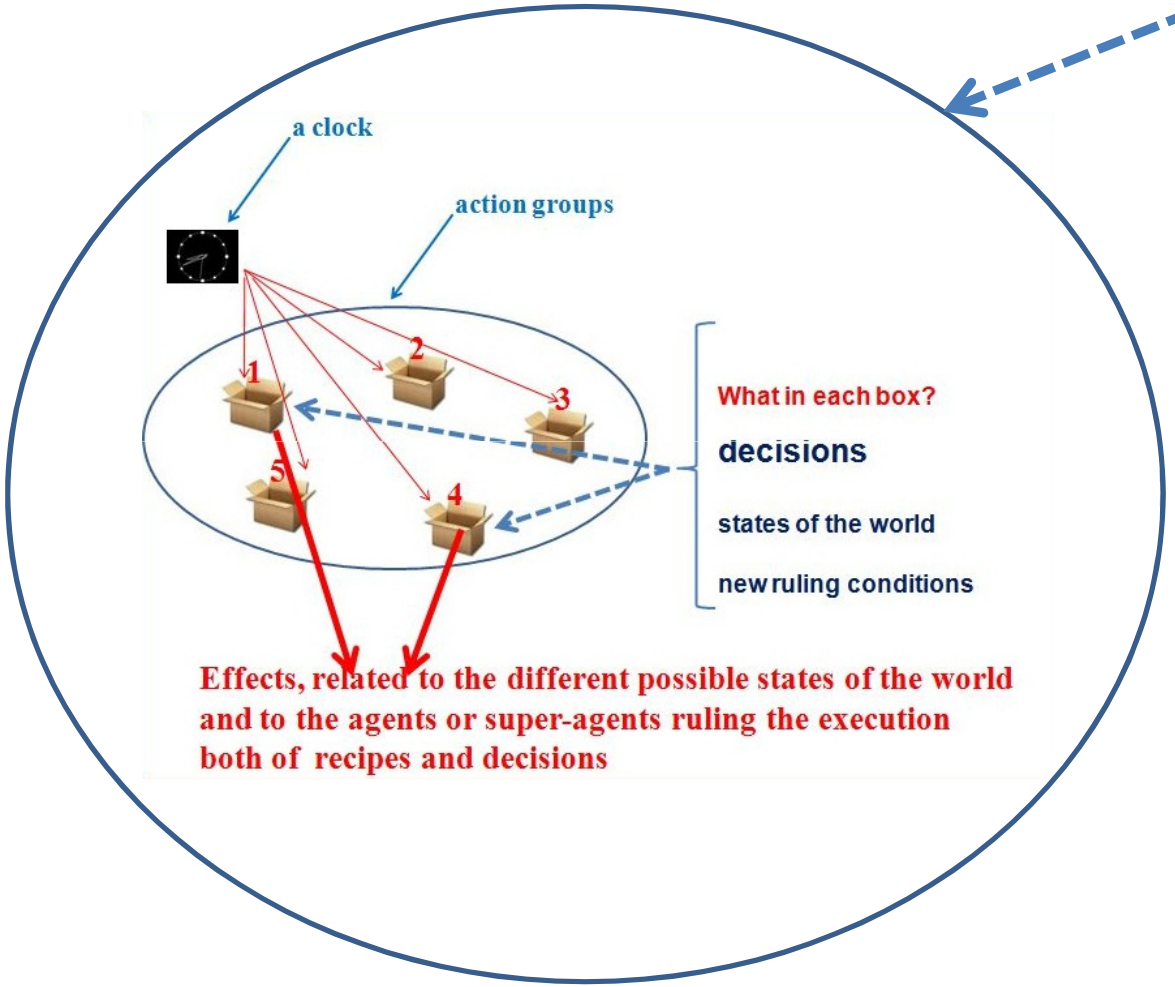
with segments to be executed in a parallel way

[[agent method] [agent method] N[agent method] [agent method] [agent method] …]
[:: [agent method] :: …]
[:: [agent method] :: …]
[:: [agent method] :: …]

with components belonging to different recipes to be executed as a whole



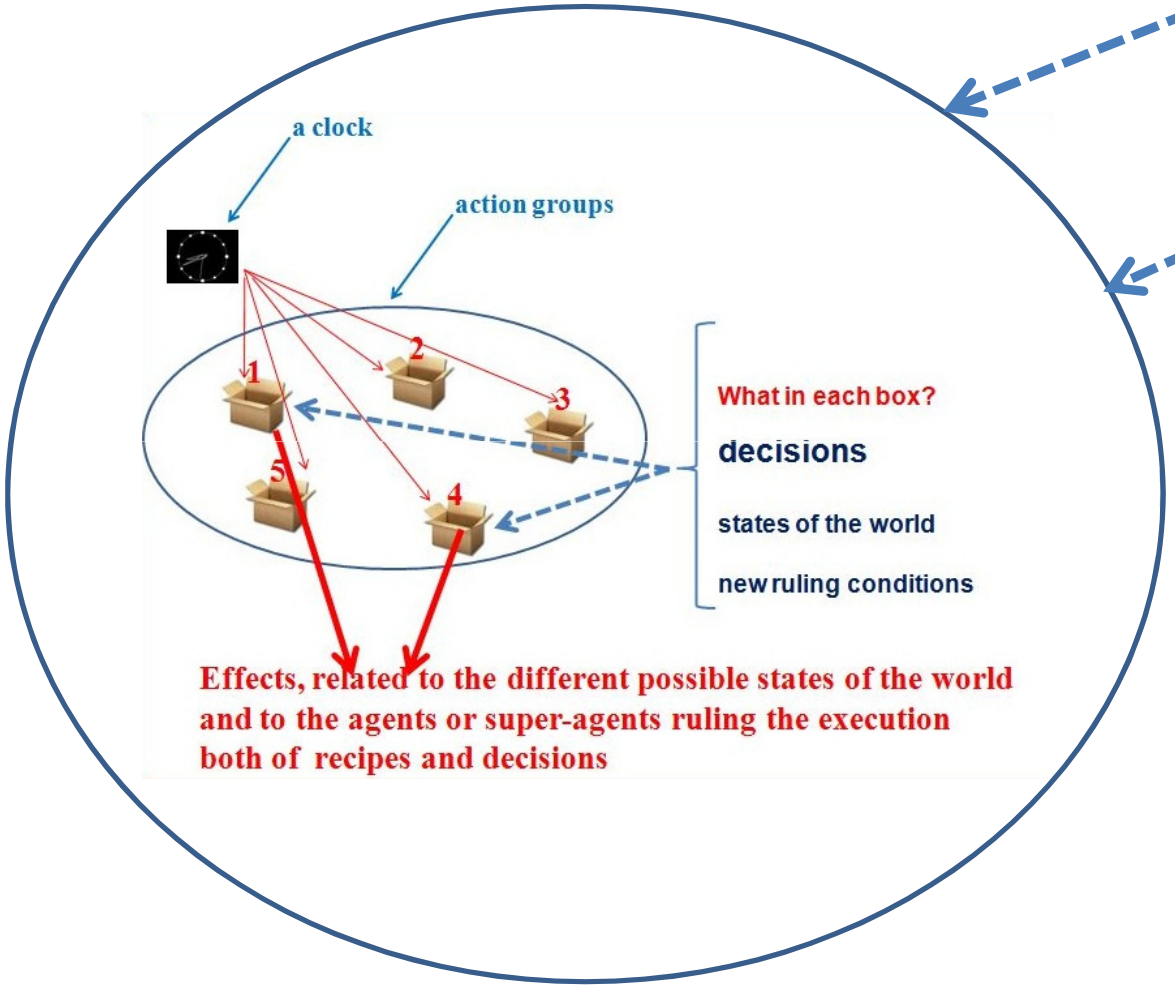
Effects, related to the different possible states of the world and to the agents ruling the execution both of recipes and decisions



repeated try and error processes with

reinforcement learning and NN to memorize and apply the outcomes

NN directly trained via a try and error process (guessing actions and effects)



Do **valid ruling** solutions emerge?

Do **anarchists** emerge?

Greatest challenges

recipes read from an external archive

[[agent method] [agent method] [agent method] [agent method] [agent method] …]

→ to be executed in a sequential way by a given crowd of agents (static or dynamic)

[[agent method] [[agent method] [agent method] [agent method]] [agent method] …]

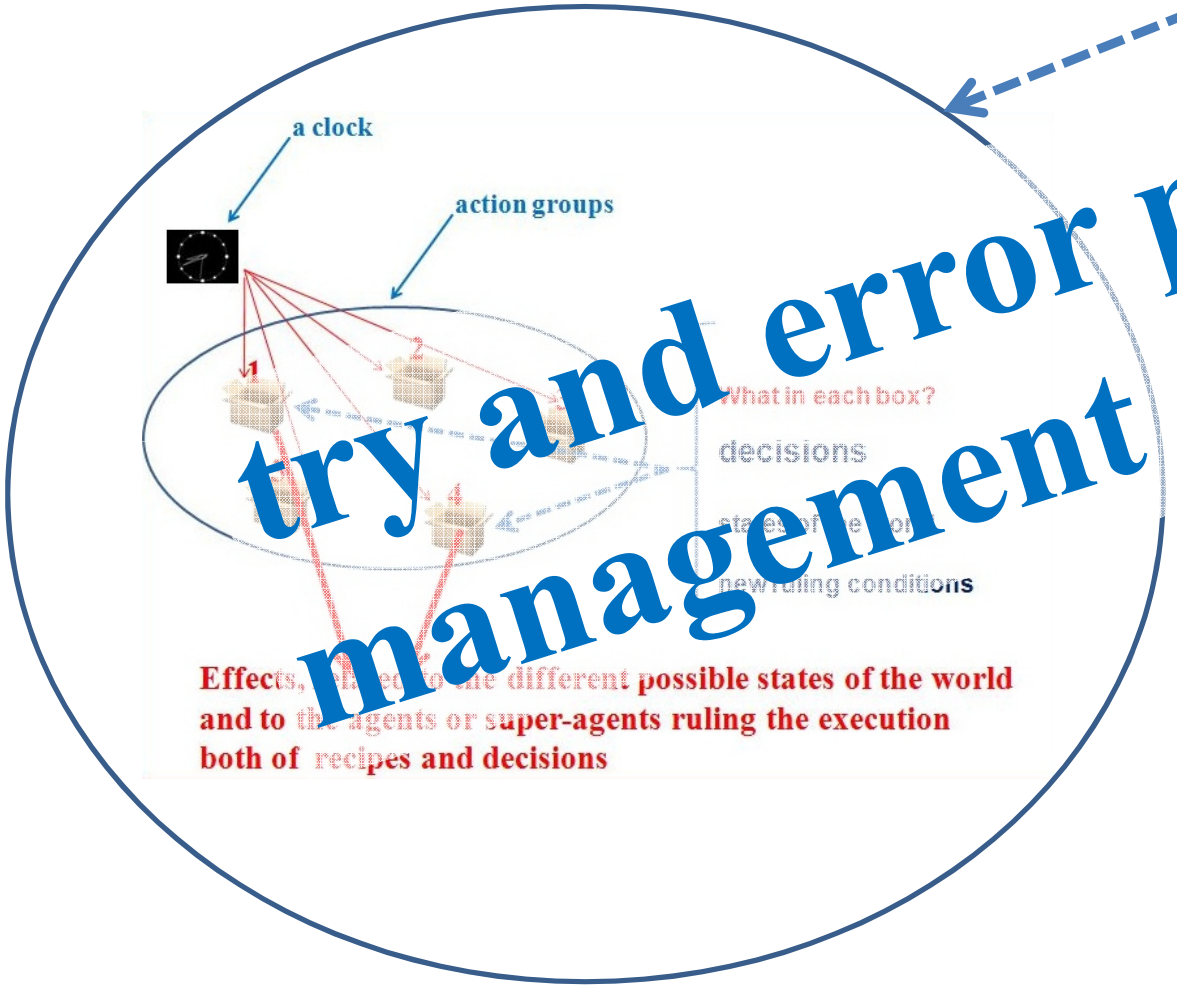
with segments to be executed in a parallel way

internal schedule management

[[agent method] [agent method] [agent method] [agent method] [agent method] …]
[:: [agent method] :: …]
[:: …]
[:: [agent method] :: …]

with components belonging to different recipes to be executed as a whole

repeated try and error processes with reinforcement learning and NN to memorize and apply the outcomes



try and error process management

NN directly trained via a try and error process (guessing actions and effects)

Thanks

Pietro Terna, terna@econ.unito.it
http://web.econ.unito.it/terna