Latsis Symposium 2012 11. - 14. Sep. | ETH Zurich, Switzerland
Economics on the Move
Trends and Challenges from the Natural Sciences

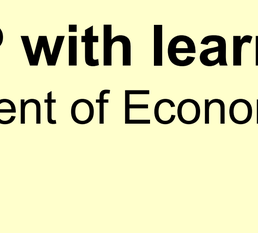
SLAPP with learning facilities: L-SLAPP

Pietro Terna, Department of Economics and Statistics, University of Torino, Italy

SLAPP, **Swarm-Like Agent Protocol in Python**, is a simplified implementation of the original Swarm protocol (www.swarm.org, 1994), choosing Python as a simultaneously simple and complete object-oriented framework.

2009, **SLAPP** basic flavor

2010, **SLAPP-Aesop**
(Agents and Emergencies for Simulating Organizations in Python)
for organization stories



2011, **0SLAPP**, with external scripts and a step toward 0MQ, for parallel computing

2012, focus on learning agents, so **L-SLAPP**

Why a new tool for **agent-based simulation?**

* For didactical reasons, applying a such rigorous and simple object oriented language as Python is.

* To use the openness of Python (python.org) and the on line interactivity and the parallelism capabilities of Ipython (ipython.org).

* **To have the possibility of using the key feature of the Swarm protocol in an easy way.**

* **SLAPP is basically the implementation of the Swarm protocol [Minar, N., R. Burkhart, C.**Langton, and M. Askenazi (1996), *The Swarm simulation system: A toolkit for building multi-agent simulations*. Working Paper 96-06-042,Santa Fe Institute, Santa Fe] **in Python.**
www.swarm.org/images/b/bb/MinarEtAl96.pdf

Reinforcement learning within an agent-based model

Agents are randomly behaving and generate a series of actions, evaluated as successful or unsuccessful via the simulation model.

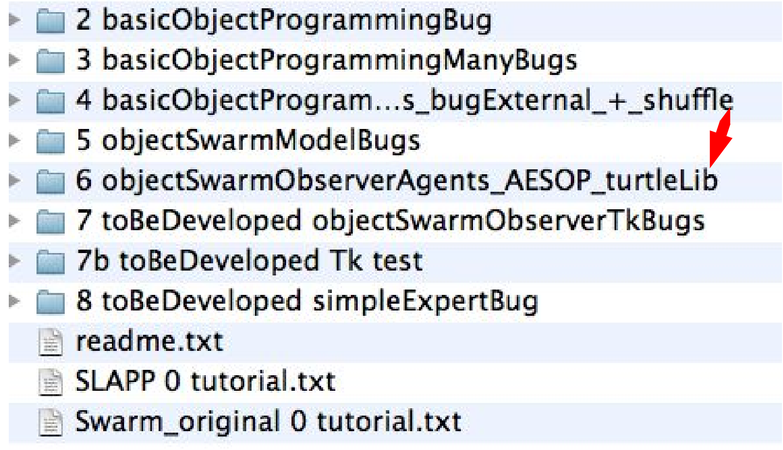
We **memorize** good and bad actions, with their ex-ante data and with the evaluation of the related effects, using one or more **neural networks**.

After the training phase, when agents have to act, they ask to the neural networks a set of guesses about the consequences of their possible actions; on this basis, they decide.

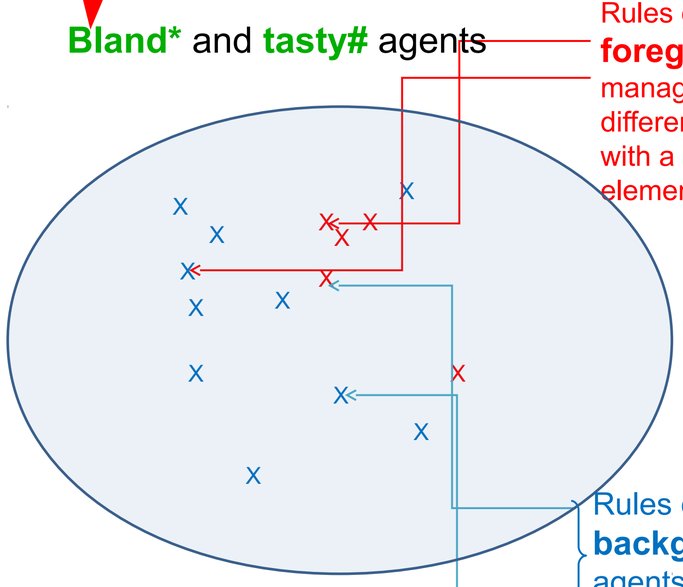
Neural network training and application are run in R (http://www.r-project.org), with Ripley's NNET function; R is connected to the Python environment, via pyRserve bridge (http://pypi.python.org/pypi/pyRserve/).

An absolutely clear and rigorous application of the *SWARM protocol* (and so, of *agent-based models*) is contained in the original SimpleBug tutorial (1996?) with ObjectiveC code files, and text files, by Chris Langton & Swarm development team (Santa Fe Institute).

In SLAPP you can find the same structure of files, but now implementing the SWARM protocol using Python



Bland* and **tasty#** agents

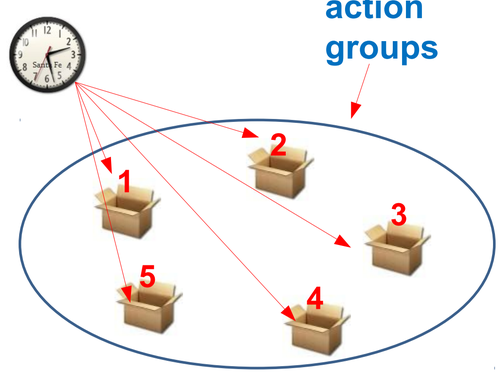


Rules operating "in the **foreground**", explicitly managed via scripts (with different sets of agents, with a different number of elements)

Rules operating "in the **background**" for all the agents, or only for the blue ones or for a specific set of tasty agents

**Bland = simple, unspecific, basic, insipid, …*
#Tasty = specialized, with given skills, discretionary, …

… with a schedule



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

recipes, as macros, read - from an external archive - the actions to be executed in a sequential way by a given thread of agents (statically or dynamically)

SLAPP with learning facilities: L-SLAPP

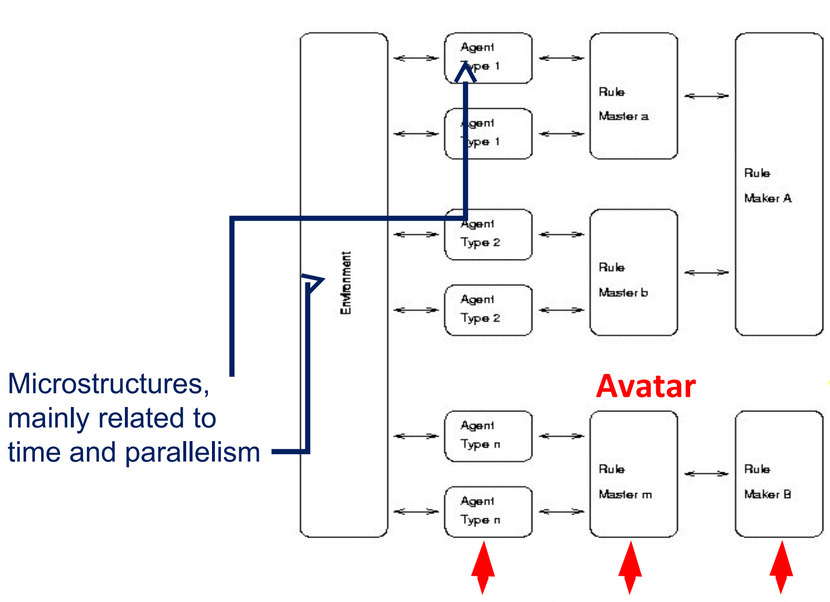
terna@econ.unito.it, web.econ.unito.it/terna, web.econ.unito.it/terna/slapp

Latsis Symposium 2012 11. - 14. Sep. | ETH Zurich, Switzerland
Economics on the Move
Trends and Challenges from the Natural Sciences

SLAPP with learning facilities: L-SLAPP

Pietro Terna, Department of Economics and Statistics, University of Torino, Italy

A generalized scheme



Fixed rules

Artificial Neural Networks (ANNs)

Classifier Systems)

Genetic Alg.

Reinforcement learning

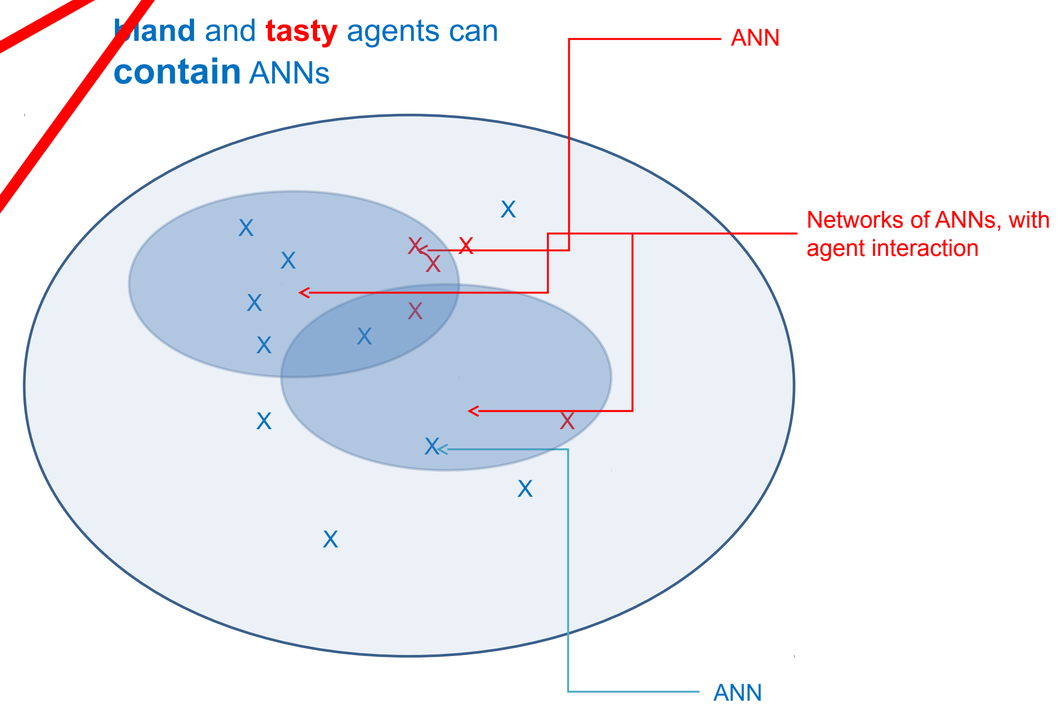
Avatar

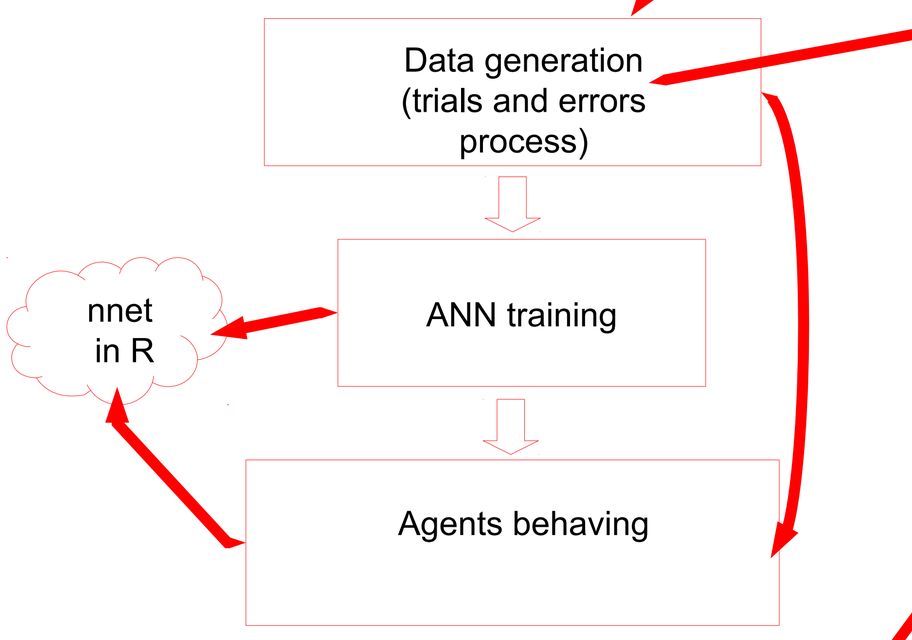
Microstructures, mainly related to time and parallelism

Agents, Rule masters, Rule makers

Artificial neural networks into the agents

Hand and **tasty** agents can **contain** ANNs





$y = g(x,z) = f(B\ f(A\ (x',z')'))$

(1) (n+m)

effect

information

action/s

or, if $z = \{z_1, z_2, \ldots, z_m\}$

$y = g(x,z) = f(B\ f(A\ (x))$

(m) (n)

an effect for each possible action

or, always if $z = \{z_1, z_2, \ldots, z_m\}$

$y_1 = g_1(x) = f(B_1\ f(A_1\ x))$

(1) (n)

…

$y_m = g_m(x) = f(B_m\ f(A_m\ x))$

(1) (n)

an ANN for the specific effect of each possible action

Artificial learning, via a trials and errors process, while acting

$y = g([x,z]) = f(B\ f(A\ (x',z')'))$ *or **p** NNs as above*

(p) (n+m)

Rule master

actions

information

effects

Coming from the simulation

Different agent, generating and using different sets, A and B, of parameters (or using the same set of parameters, as collective learning)

Emergence of new *norms* **[modifying U=f(z) , as new norms do]** and *laws* **[modifying the set y, as new laws do]**

the agents will choose y maximizing:

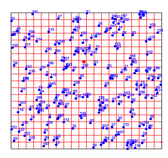
(i) individual U, with norms

(ii) societal wellbeing

accounting for social norms

The learning side (L-) of SLAPP:

nnet&reinforcementLearning



under development (taking it as possible easy)

look at the file **z_learningAgents_v.?.?.zip**
(Python 2.7.x) at

goo.gl/SBmyv

we need Rserve running; instruction at
goo.gl/zPwUN

SLAPP with learning facilities: L-SLAPP

terna@econ.unito.it, web.econ.unito.it/terna, web.econ.unito.it/terna/slapp