Swarm Intelligence Course (INFO-H-414)
Exams

April 23, 2015

1 Modalities

The exam is divided in two parts:

Project You will be assigned a project among the ones proposed below. You are asked to provide the required deliverables and to present your project in a 7-minute talk, followed by 5 minutes of questions. This will account for up to 10 points of your final grade.

Questions You will be asked a number of questions concerning *the entire course material*. This will account for up to 10 points of your final grade.

To pass the exam one must collect at least 5 points for each part of the exam.

1.1 Timeline

- The date of the exam is still to be decided.
- The project submission deadline is 5 working days before the date of the exam (for instance, if the exam date would be fixed on June 10th, then the project is due for June 3rd at 23.59).
- Each day of delay on the submission will entail a penalty of 2 points on the final evaluation of the project.

1.2 Project Deliverables

For the project, you will have to provide:

- Your code in digital format (i.e., text files), so we can test it. Send it by e-mail to the people responsible for your project (see contacts at the end of the document);
- A short document (6-8 pages) written in English that describes your work. You have to explain both the idea and the implementation of your solution.

- On the day of the oral presentation, you are expected to show slides and come with your own laptop. If you happen to not have a laptop, send us a PDF version of your slides **before** the day of the exam.

2 Projects

2.1 General Remarks

Apply what you learnt It is mostly important that you stick to the principles of swarm intelligence: simple, local interactions, no global communication, no global information.

Avoid complexity If your solution gets too complex, it is because it is the wrong one.

Honesty pays off If you find the solution in a paper or in a website, cite it and say why you used that idea and how.

Cooperation is forbidden Always remember that this is an **individual** work.

The project counts for 50% of your final grade. The basic precondition for you to succeed in the project is that it must work. If it does not, the project won't be considered sufficient. In addition, code must be understandable — comments are mandatory.

The document is very important too. We will evaluate the quality of your text, its clarity, its completeness and its soundness. References to existing methods are considered a plus — honesty *does* pay off! More specifically, the document is good if it contains all the information needed to reproduce your work without having seen the code and a good and complete analysis of the results.

The oral presentation is also very important. In contrast to the document, a good talk deals with *ideas* and leaves the technical details out. Be sure that it fits in the 8-minute slot.

2.2 Ant Colony Optimization

2.2.1 Introduction

This project comprises the design, implementation and analysis of ACO algorithms for solving the *capacitated vehicle routing problem* (CVRP). The CVRP belongs to the class of vehicle routing problems (VRP), which are classical NP-hard combinatorial optimization problems. In this problem, a supplier of a given product has a list of n customers, each with a specific demand q_k. The supplier must meet the demand of each client, using a set of identical vehicles that depart from the depot and must return to it. Additionally, the vehicles present a maximum transportation capacity Q, and going from one city to another presents a transportation cost. To solve this problem, an algorithm must find the number of vehicles m and a route for each of these vehicles so that

the total transportation costs considering all vehicles is minimized. A detailed description of the CVRP can be defined as follows:

a. Input

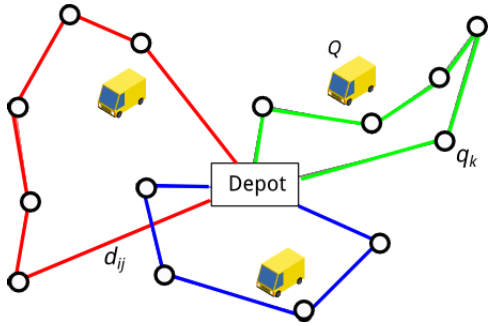
- Traveling cost (d_{ij}) – the matrix d_{ij}, $i, j = 0, \ldots, n$, defines the cost of going from location i to location j; $i, j = 0$ corresponds to the depot.
- Demand (q_k) – each customer $k = 1, \ldots, n$ has a demand q_k.
- Maximum vehicle capacity (Q) – the maximum capacity of the vehicles (all vehicles have the same maximum capacity).

b. Objective: to determine the number of vehicles m and the routes they will follow (subset of ordered customers) so the total transportation cost is minimized.

c. Constraints:

- All vehicles are initially located at the depot.
- Each vehicle cannot serve a route with demand bigger than Q.
- Each customer is served by only one vehicle.
- The routes of the vehicles must begin and end at the depot.

The following figure gives an example of a solution for the CVRP, the total transportation cost is then calculated as the total travel cost of all the vehicles.



2.2.2 Goal

The goal of this project is to design, implement and analyze ACO algorithms for solving CVRP. For doing so, your task is to adapt and extend two of the ACO algorithms that you have studied during the exercise sessions:

- Max-min Ant System (MMAS)
- Ant Colony System (ACS)

Note that since the algorithms will be designed for a different problem (CVRP), modifications will be needed. You are free to define the components of the algorithms as you think best, as long as they follow the general definition of each algorithm. You are allowed to implement ideas proposed in the literature, as long a citation is included. Once the algorithms are implemented, experiments should be carried out in order to compare and analyze their performance. Later, the addition of local search the ACO algorithms should be studied. You are free to propose any kind of local search for the problem.

We encourage you to search for related literature to inspire your algorithms. An extense bibliography on this problem can be found at http://neo.lcc.uma.es/vrp/bibliography-on-vrp/. In addition, we provide a set of instances for the experiments. You can download them in http://iridia.ulb.ac.be/~lperez/INFO-H-414/project/instances.tar.gz. These instances were obtained from the Augerat data sets, which you can find in [4] and [1], as well as other data sets. A description of the instance files can be found in the tsplib-descr.ps file. The format is similar to the one of the TSP instances used in the class exercises, and so you can adapt the code provided in the template. For the tuning, we provide a set of training instances (if you are using manual tuning, disregard this).

2.2.3 Deliverables

The final deliverables include the source codes and documents. As a guideline we provide the points assigned to the report and the code. Notice that these points do not account for all the project given that we evaluate also the final presentation.

- Report (18 points)
  a. Implement MMAS and ACS for solving the CVRP (10 points).
     (a) Description of the implemented algorithms. Please explain your design decisions (representation of solutions, pheromone and heuristic information definition, update mechanism, transition rule, etc.).
     (b) Tune the algorithms using the provided training set and report the best parameter settings for each algorithm. Describe the process you used to obtain the parameter settings. If you used automatic tuning, please report: parameters, set of values and number of evaluations allowed to the tuner (some freely available tuners are [3] and [5]).
     (c) For each algorithm, perform 10 runs on each instance of the test set. (Note: for comparison purposes you must set the same budget (number of evaluations) for both algorithms.)
     (d) Perform a pairwise comparison of the algorithms using the Wilcoxon rank-sum test. Report for each instance/algorithm experiment:

best (B), worst (W), mean (M) and standard deviation (SD). Provide plots of your results.

(e) Based on d), which of the implemented algorithms would you recommend for solving this problem? Comment.

(f) Compare the convergence of the algorithms. (Note: The goal is to compare the computation effort (evaluations) vs. solution quality).

b. Choose an algorithm (MMAS or ACS) and add local search search to it (8 points).

(a) Describe the local search procedure implemented.

(b) Perform 10 runs per instance for the new algorithm and compare it with its version without local search. Report as above in d).

(c) Are there any improvements? Why? Comment.

c. Optional 1: Tune the parameters of the algorithm chosen in 2). Analyse your results, is the algorithm more sensitive to some parameters? (2 points)

d. Optional 2: Compare your results with the ones you can find in [2]. Notice that the instances used in that work are different from the ones used in this project, and can be found at `http://people.brunel.ac.uk/~mastjjb/jeb/orlib/files/` (files vrpnc*.txt). Additional parameter tuning might be necessary! (2 points)

- Code (2 points)
  - The code should run in the same way as the code developed in class, that is, using the same command line arguments. Feel free to add new arguments in case you need them.
  - The code should be properly commented and ordered (indent!).
  - The code should include a README file with the main instructions and specifications of your algorithms and parameters.
  - You must implement your own code. Plagiarism will be penalized.

Describe your algorithms in detail, the aim of this project is to evaluate your ability to apply the ideas behind of the ACO algorithms and therefore a crucial step is the definition of the algorithm components. The results of your experiments are as important as the analysis you make out of them. Please do not report results without commenting and interpreting them.

2.3 Swarm Robotics: *Collective Decision Making with Heterogeneous Agents*

Decision making is a cognitive capability which allows individuals to select the best option among a set of possible alternatives. When a decision is taken by a group of agents, the process is defined as collective decision making.

Collective decision making is a fundamental cognitive capability that can be found in several natural systems. For instance, each spring, honeybee colonies collectively tackle as a decision problem the nest site selection. A different example is the collective decision of bird flocks, which collectively move in one direction without need of any leader.

In this project, the student is asked to provide a robot swarm with the capability of discriminating between four options and collectively select the best one. The robot swarm is composed of heterogenous robots, that is, robots have different types of sensors which allow them to perceive only a subset of the quality features characterising an option. While each robot has partial knowledge of the available alternatives and can only partially estimate each option's quality, the swarm, as a whole, can perceive the entire set of features and is able to collectively decide for the best option. Therefore, robots must explicitly collaborate to agree on the selection of the best alternative.

2.3.1 Problem definition

The robot swarm operates in a scenario composed of 4 *target* rooms r_i (with $i \in [0, 3]$) connected by a central room. Each target room has a quality value v_i (with $i \in [0, 3]$), which is computed as the average of three metrics:

$$v = (v_G + v_L + v_O)/3 \tag{1}$$

The three metrics are computed as function of the following environmental characteristics:

- **The ground color (v_G)**: the ground color varies in the full gray scale from white to black and determines the value $v_G \in [0, 1]$. We assume that $v_G = 1$ for white grounds, $v_G = 0$ for black grounds and in the between v_G scales linearly. The robots can perceive the ground color through the ground sensor.

- **The light intensity (v_L)**: each room has a different illumination which determines $v_L \in [0, 1]$. The room's light intensity is determined by the strength of a light source placed in the center of each room. We assume that $v_L = 1$ for light sources with maximum light intensity and $v_L = 0$ for light sources with minimum intensity (i.e., no light) and in the between v_L scales linearly. The robots can perceive the light intensity through the light sensor.
  Note: the light sensor returns a measure that varies with respect to the distance from the light source. Therefore, a robot may integrate several sensor readings over time while moving in the room for estimating an average light intensity of the room.

- **The number of objects (v_O)**: each room stores a number of objects $\mathcal{O}_i \in [2, 12]$ which determines $v_O \in [0, 1]$. We assume that v_O is linearly proportional to the number of objects $\mathcal{O}$, i.e., the more the better. Each object is marked with a green LED, which can be perceived by the robots through the camera.

Goal The robot swarm has to select the room r_i with best quality v_i:

$$r_i,\ i \in \arg\max_{i \in [0,3]} v_i \tag{2}$$

To select a room a robot must physically move into the selected room. We assume that the swarm has done a collective decision when the quorum $Q = 0.9$ of the total swarm moved into a room r. For instance, for a swarm of 100 robots, the decision is considered as taken when at least 90 robots have moved into the same target room.

Heterogeneity The swarm, in the given configuration, is composed of 20 heterogenous robots of two types:

- **Type G**: 10 robots of type G are equipped with a ground sensor that allows them to sense the color of the ground. However, robots of type G do not have sensors to perceive the light intensity value.
- **Type L**: 10 robots of type L are equipped with a light sensor that allows them to sense the light intensity. However, robots of type L are not equipped with the ground sensor, thus cannot perceive the color of the ground.

Robots of both types are equipped with camera and therefore are able to sense and count the objects (marked with green LED) stored in each room.

Additional Remarks

- The environment is designed with a central room connecting the 4 target rooms, so that the robots can encounter in the central room and interact with each other.
- Robots are able to differentiate the target rooms from each other by mean of LED lights of different colours that are placed on the door of each target room. The colours are assigned as follows: magenta, blue, orange, and red for rooms 0,1,2,3 respectively. The robots can also use these lights to localise themselves in the environment.
- The student can develop a single lua script that checks whether the robot is of type G or L. This can be done either by checking if a sensor is present or not (not null or null), or by simply checking the id of the robot. Robots of type G are assigned an id prefix "fbG", while the prefix of robots of type L is "fbL".
- The student can use the LEDs actuators to visualise the type of each robot, e.g. yellow LEDs for robots with ground sensors and cyan LEDs for robots with light sensors (remember to not use the same colours of rooms and objects).

Figure 1 shows a top view of the environment composed of four target rooms and a central room. The central room has no light and black ground. Each room has a different quality which may vary in each experiment and is determined by the composition of the three metrics v_G, v_L and v_O (see Equation (1)). The quality of the four target rooms of the example illustrated in Figure 1 are given in the Table 1. The best is room is room number 2 with the evaluation: $v_2 = 0.531857$.

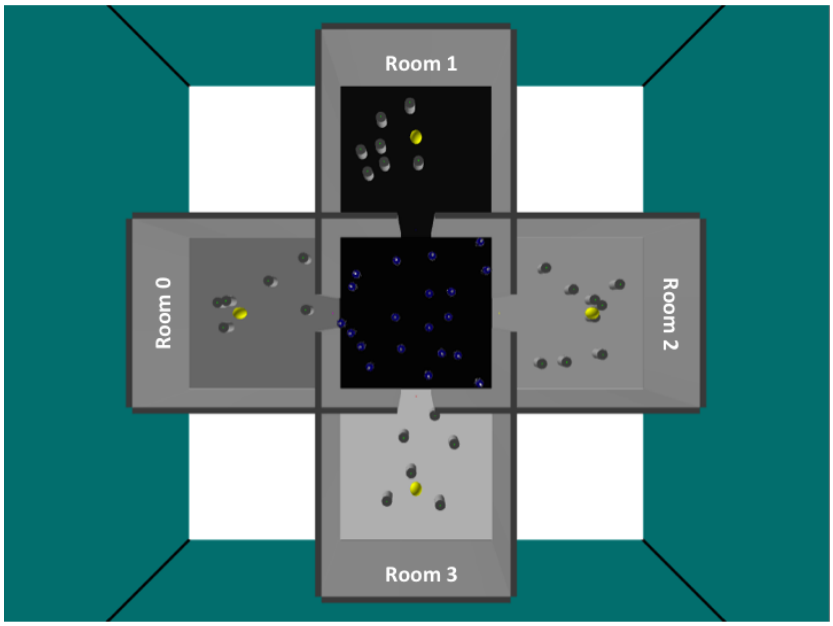


Figure 1: A top view of the environment.

Room i	v_G	v_L	v_O (O)	**Room quality** v_i
Room 0	0.478243	0.579924	0.4 (6)	0.486056
Room 1	0.050487	0.72393	0.5 (7)	0.424806
Room 2	0.609765	0.285806	0.7 (9)	0.531857
Room 3	0.743975	0.386149	0.4 (6)	0.510041

Table 1: Target-room's quality values of the example depicted in Fig 1.

2.3.2 Goals

The goal of this project is to design, implement and test a robot controller that allows the swarm to collectively select the best room r_i between 4 possible target rooms. The best room has the the highest quality v_i.

Different solutions are possible. Bonus points are awarded for simple and reactive controllers with limited (or no) memory. Remember to follow the principles of swarm robotics and apply what you learned during the course. Solutions in which the robots aggregate in a database global information about every room will be considered poor.

The student is required to analyse the solution at first with the provided configuration, and after by varying the following parameters:

- The swarm size N in the range of values $\{25, 30, 35, 40\}$ robots.
- The ratio $\rho = T_G/(T_G + T_L)$ where T_G is the size of the subpopulation of robots of Type G, and T_L is the size of the Type L robot subpopulation. In the initial configuration, $\rho = 0.5$ with the swarm splited in two equal halves of the two types. The student is asked to analyse how varying ρ affect the system.

A complete analysis must be performed to evaluate the quality of the behaviour. In particular, quantitative numerical measures of the performance of the system must be produced. ARGoS automatically dumps data on a file whose name must be set by you in the XML experiment configuration. This file is composed of several lines, each line containing five elements:

- The current step
- The number of robots in room 0
- The number of robots in room 1
- The number of robots in room 2
- The number of robots in room 3

To present statistically meaningful results we suggest that you repeat your experiments at least 30 times.

Be aware that, for the project evaluation, **the analysis is as important as the implementation**. A project presenting a behaviour that is capable of selecting the optimal room wonderfully will be evaluated poorly if the analysis part is missed or limited.

Technical information on ARGoS, Lua, and the experiment configuration file is reported in the swarm robotics page of the course: http://iridia.ulb.ac.be/~lgarattoni/extra/h-414/.

Setting up the code

- Download the experiment files: SR_Project_H414.tar from http://iridia.ulb.ac.be/~lgarattoni/extra/h-414/SR_Project_H414.tar.
- Unpack the archive and compile the code:
  - $ tar xvf SR_Project_H414.tar # Unpacking

  - $ cd SR_Project_H414 # Enter the directory
  - $ mkdir build # Creating build dir
  - $ cd build # Entering build dir
  - $ cmake -DCMAKE_BUILD_TYPE=Release ../src # Configuring the build dir
  - $ make # Compiling the code

- Set the environment variable ARGOS_PLUGIN_PATH to the full path in which the build/ directory is located:

  $ export ARGOS_PLUGIN_PATH=/path/to/SR_Project_H414/build/

- You can also put this line into your $HOME/.bashrc file, so it will be automatically executed every time you open a console. Run the experiment to check that everything is OK:

  - $ cd /path/to/SR_Project_H414 # Make sure you are in the right directory
  - $ argos3 -c decision-making.xml # Run the experiment

If the usual ARGoS GUI appears, you're ready to go.

2.3.3 Deliverables

The final deliverables must include source code and documentation:

Code: The Lua scripts that you developed, well-commented and well-structured.

Documentation: A report of 6-8 pages structured as follows:

- Main idea of your approach.
- Structure of your solution (the state machine).
- Analysis of the results.

3 Contacts

Marco Dorigo	mdorigo@ulb.ac.be	for general questions
Mauro Birattari	mbiro@ulb.ac.be	for general questions
Leslie Pérez Cáceres	leslie.perez.caceres@ulb.ac.be	for ACO
Leonardo Bezerra	leonardo@iridia.ulb.ac.be	for ACO
Andreagiovanni Reina	areina@ulb.ac.be	for swarm robotics
Lorenzo Garattoni	lgaratto@ulb.ac.be	for swarm robotics

References

[1] Branch and Cut.org. Data set vrp. http://www.coin-or.org/SYMPHONY/branchandcut/VRP/data/index.htm.

[2] K. Doerner, M. Gronalt, R. Hartl, M. Reimman, C. Strauss, and M Stummer. Savingsants for the vehicle routing problem. In Cagnoni et al., editors, *EvoWorkshops02*, volume 2279 of *Lecture Notes in Computer Science*. Springer-Verlag, 2002.

[3] Irace. Iterated race for automatic algorithm configuration. http://iridia.ulb.ac.be/irace/.

[4] Networking and Emerging Optization. Vrp website. http://http://neo.lcc.uma.es/vrp/.

[5] Paramils. An automatic algorithm configuration framework. http://www.cs.ubc.ca/labs/beta/Projects/ParamILS/.