Internal Report 2012-13 November 2012

Universiteit Leiden
Opleiding Informatica

Enhancing medicine with nanotechnology:

a simulation study

Francesco Bigarella

Supervisors:
Prof. Dr. Thomas Bäck
Dr. Michael Emmerich

MASTER'S THESIS

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

"Show me how you do that trick
The one that makes me scream" she said
"The one that makes me laugh" she said
And threw her arms around my neck
"Show me how you do it
And I promise you I promise that
I'll run away with you"

— The Cure, Just like heaven

Dedicated to my grandfather.

ABSTRACT

This document tries to define and analyze a new discovery in medical science: nanorobots.

With nanotechnology the way a patient is cured and monitored is going to change drastically and we are trying here to define how nanorobots will achieve this result.

The number of tasks nanorobots can do is still to be defined because it strictly depends of the improvements in nanotechnology: repair tissues, clean blood vessels and airways, monitor the body functions are just some of the operations a swarm of nanorobots can perform.

The purpose of this thesis project is to examine and evaluate the available literature about nanomedicine and develop a simulation based on that knowledge. To do so a model for the nanorobots has been defined, in which we choose each component and give priority to those that have already been fabricated or are likely to be created in the near future. Delineated the model, two software modules have been developed, both based on the NetLogo multi-agent framework: one program used to define the input environment for the simulation and the other to run it, allowing the study of swarm behaviour.

The objective of the nanorobots was to find and destroy a designed target, identified in the figure of a cancer cell lying in a blood vessel. To do so a swarm computation technique named Artificial Bee Colony (ABC) has been modified and implemented.

Tests results show how such task can be successfully achieved but it requires a good balance between the number of agents and the amount of medical drug each one equip. The use of chemical signals for communication has been proved useful but only locally, it is not effective for the coordination of a swarm scattered in the environment.

Keywords
nanorobots, nanomedicine, swarm computation, swarm intelligence

Grazie per avermi dimostrato che, per cambiare tutto,
basta solo un pizzico di coraggio e...
un posto meraviglioso come questo!

— Noemi De Pasquale

ACKNOWLEDGMENTS

At this point of my academic career I feel I have to take a moment and think.
Think about all the people that helped me to get here.
My family, aunt and cousins.
Think about all the people that contribute to this thesis and my formation.
Prof. Bäck, Prof. Gaggi and all the people at the University of Padova and Leiden.
Think about all the people I have shared so many moments with, nice and less nice.
My friends, old and new.

Thanks everybody, I am here because of you.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LISTINGS

ACRONYMS

RBCs	Red Blood Cells or erythrocytes
BP	Blood Pressure
ACO	Ant Colony Optimization
PSO	Particle Swarm Optimization
ABC	Artificial Bee Colony
MAS	Multi-agent System
PEAS	Performance, Environment, Actuators, Sensors
CoMa	Robots Communication Marker
CCT	Cancer Chemical Trace
DT	Drug Trace
RFID	Radio Frequency Identification Device

Part I

RESEARCH & MODELLING

1 INTRODUCTION

1.1 A FANTASTIC VOYAGE

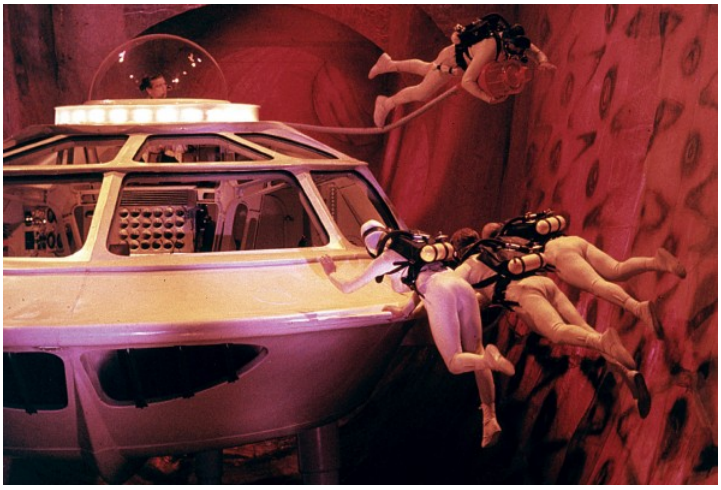


Figure 1: Still of the film *Fantastic Voyage*, 1966.

Year 1966. Scientist Jan Benes, who knows the secret to keeping soldiers shrunken for an indefinite period, escapes from behind the Iron Curtain with the help of CIA agent Grant. While being transferred, their motorcade is attacked. Benes strikes his head, causing a blood clot to form in his brain. Grant is ordered to accompany a group of scientists as they are miniaturized. The crew has one hour to get in Benes's brain, remove the clot and get out [15]. This is the beginning of the science fiction cult film: Fantastic Voyage.

Many years have passed since the storyline for the film was written, and that idea is still far from reality. But not by much.

All that was science fiction before is now known as NANOROBOTICS. It can be defined as an emerging technology creating machines or robots whose components are at or close to the scale of a nanometer (10^{-9} meters). More specifically, nanorobotics refers to the still largely theoretical nanotechnology engineering discipline of designing and building nanorobots (also known as nanobots, nanoids or nanomachines).

The birth of Nanotechnology is usually associated with a talk by Nobel-prize winner Richard Feynman entitled "There is plenty of room at the bottom", whose text may be found in [11]. Nanotechnology has the potential for major scientific and practical breakthroughs. Nanomachines can indeed be employed in countless useful applica-

tions, first of all in medical technology, which could be used to identify and destroy cancer cells. Another potential application is the detection of toxic chemicals, and the measurement of their concentrations, in the environment.

1.1.1 *Nanomedicine*

For many years, medical instruments have been rapidly developed so that diagnosis and treatment can be done more effectively and efficiently. Nevertheless, the diseases continuously develop themselves to counteract the treatment and new diseases are discovered.

Nanorobotic technology can plausibly play a crucial role in biomedical engineering.

The aim of nanorobotic research in medical field is to use nanorobots as an alternative medical instrument. Due to its small size, a nanorobot can travel through human blood vessels network to the target within a blood vessel directly; thus, nanorobots can potentially be used in a drug delivery system to transport drug to a specific target area instead of using traditional way such as injection that may cause undesired effects to other areas. In the future, nanorobots may become a solution for many currently incurable diseases such as cancer by delivering anti-cancer drug to kill cancer cells directly without doing any harm to the normal cells [33]. Consequently, side effects of anti-cancer drugs which are the major problem of current cancer therapies can be reduced. Artificial cells (nanorobots) can also be used to patrol the circulatory system, detect small concentrations of pathogens and destroy them. This would amount to a programmable immune system, and might have far reaching implications in medicine, causing a paradigm shift from treatment to prevention.

Nanomedicine seeks to deliver a valuable set of research tools and clinically useful devices in the near future. The National Nanotechnology Initiative[1] expects new commercial applications in the pharmaceutical industry that may include advanced drug delivery systems, new therapies, and in vivo imaging. Further down the line, the speculative field of molecular nanotechnology believes that cell repair machines (molecular machines capable of entering the cell and repair it) could revolutionize medicine and the medical field.

Nanomedicine is a large industry, with nanomedicine sales reaching 6.8 billion dollars in 2004, and with over 200 companies and 38 products worldwide, a minimum of 3.8 billion dollars in nanotech-

1 The National Nanotechnology Initiative is a United States federal nanoscale science, engineering, and technology research and development program. More info at http://nano.gov/about-nni

nology R&D is being invested every year [43]. As the nanomedicine industry continues to grow, it is expected to have a significant impact on the economy.

1.2 PROBLEM DESCRIPTION

The objective of this Master Thesis was to study the existing literature about nanorobotics and find a plausible solution for the problem of destroying harmful cells inside the human circulatory system. Many possibilities of application exist for nanomedicine, but we will consider for this study only articles focusing on the cure of cancer.

The first task was to summarize and comprehend all the existing work and, based on that, develop a model for a nanorobot that can be really realized in a few years with the available technology. Since the filed of study is really recent (almost all the scientific literature starts after 2000) and the practical application of the study still not possible, the number of articles and publications on the subject is quite low. It was therefore necessary to spend more time searching for related articles (such as general nanorobotics and biology) to fill this gap of sources.

Because of the impossibility to implement a technology as this in a real experiment, the only option available was to build a simulated version of a real-world application and use is as a test environment for nanorobots: this made it possible to study the effectiveness of the considered solutions and the collaborative behaviour of nanorobots.

From the literature, indeed, it is anticipated that swarm intelligence techniques inspired by social animals and insects in nature could lead to effective, robust control of a swarm of nanorobots. Nevertheless, using nanorobot swarm in medical applications, nanorobots must operate in highly dynamic environment inside human bodies; the expected environment should consequently be taken into account in modelling nanorobots. In many medical applications, nanorobots operate in the cardiovascular system that passes blood cells throughout the body. In this study different approaches of nanorobots design were considered, together with problems that can arise using such models.

The goal of the project was to study the behaviour of a swarm of simple agents coordinating to perform a given task, using swarm intelligence paradigms. The task given was to search for a target element, collocated in the environment by the user, and destroy it using the drug stored inside every nanorobot. Nanorobots had to have a communication system and a way to navigate in the environment. It was important to always follow the swarm golden rule: every agent

is the simplest possible and together they can perform complex tasks.

Under the supervision of Prof. Dr. Thomas Bäck[2] from the LIACS[3] a work plan was defined, dividing the project in different steps:

1. Preliminary literature research of scientific and technical publications (articles, journals, essays, master and phd thesis) about the topic in various digital libraries [36][23][31][42];
2. Feasibility study and definition of the problem;
3. Search for tools, resources and libraries to use for the implementation of the simulation;
4. Definition of a model for the nanorobot and the environment;
5. Coding the simulation;
6. Testing the simulation, studying the results obtained.

1.3 DOCUMENT STRUCTURE

This report consists of 8 chapters and one appendix.

Chapter 1 introduces the project. Chapter 2 examines the environment the nanorobots will work in, the human circulatory system. Chapter 3 explains the state of the art of nanorobots technology and all related ideas and concepts. Chapter 4 describes the swarm computation theory and how it can be applied to solve the coordination problem in our study. Chapter 5 describes the defined model for the nanorobots and the environment. Chapter 6 and 7 describe the simulator the simulations that were made and the results obtained. Chapter 8 concludes the report and the three appendixes present respectively the used tool, the implemented code and screenshots of the simulator.

2 Prof. Dr. T.H.W. (Thomas) Bäck, University of Leiden, http://www.liacs.nl/organization/people/showdetails?ID=4

3 Leiden Institute of Advanced Computer Science (LIACS), University of Leiden, http://www.liacs.nl

2

THE HUMAN CIRCULATORY SYSTEM

2.1 INTRODUCTION

The circulatory system is an organ system that passes nutrients, gases, hormones, blood cells and more elements to and from cells in the body in order to maintain them healthy and functional, stabilize body temperature and pH, and to help fight diseases.

This system may be seen as a blood distribution network, but the truth is that the circulatory system is composed of the cardiovascular system, which distributes blood, and the lymphatic system, which returns excess filtered blood plasma from the interstitial fluid (between cells) as lymph. Humans have a closed cardiovascular system, meaning that the blood never leaves the network of arteries, veins and capillaries which content, the blood, does not directly come in contact with the cells and tissues in the body to deliver the nutrients, task assigned to the lymphatic system.

Main components of the human cardiovascular system are the heart, blood, and blood vessels. Such system includes: the pulmonary circulation, a "loop" transporting blood from the right ventricle to the lungs, where blood is oxygenated, and back to the left atrium; and the systemic circulation, a "loop" carrying blood from the left ventricle to the tissues in all parts of the body and then returns the blood to the right atrium.

2.2 BLOOD

Blood is a specialized bodily fluid in animals that delivers necessary substances such as nutrients and oxygen to the cells and transports metabolic waste products away from those same cells.

It is composed of blood cells suspended in a liquid called blood plasma. Plasma is mostly water, and contains many dissipated entities such as proteins, glucose, hormones and carbon dioxide and blood cells themselves. The blood cells are mainly Red Blood Cells or erythrocytes (RBCs), white blood cells and including platelets (Figure 2).

While red blood cell main job is to carry oxygen to the body tissues, white blood cells help to resist infections and parasites (as part of the

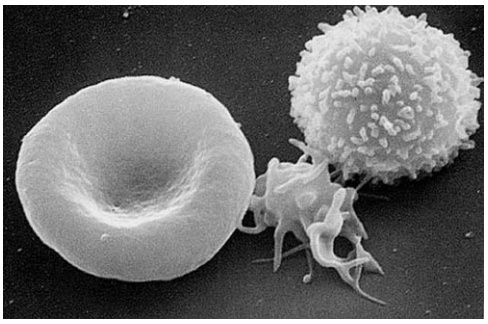


Figure 2: A scanning electron microscope (SEM) image of a normal red blood cell, a platelet, and a white blood cell[1].

human immune system) and platelets are part of the blood coagulation process.

Humans contain about 5 litres of blood, accounting for 7% of body weight. Red blood cells constitute about 45% of this volume and white blood cells about 1%, the rest being the liquid blood plasma (data from [1]).

2.3 BLOOD VESSELS

A blood vessel can be considered a channel or conduit through which blood is distributed to body tissues. Based on their structure and function, blood vessels are classified as either arteries, capillaries or veins.

They have different structures, veins having two layers, arteries three and capillaries just one thin permeable layer of cells.

The blood pressure in blood vessels is expressed in millimetres of mercury (1 mmHg = 133 Pa). In the arterial system it is usually around 120 mmHg and 80 mmHg and in the venous system is almost constant and rarely exceed 10 mmHg [12].

Blood vessels play a huge role in virtually every medical condition. Cancer, for example, can not progress unless the growing tumour receives nutrients provided by the blood. Another example is atherosclerosis, the formation of lipid lumps in the blood vessel wall, and the probably most common cardiovascular disease.

2.3.1 *Arteries*

The arteries are blood vessels that carry blood away from the heart and distribute it to the various tissues of the body. This blood is nor-

1 Image available at http://web.ncifcrf.gov/

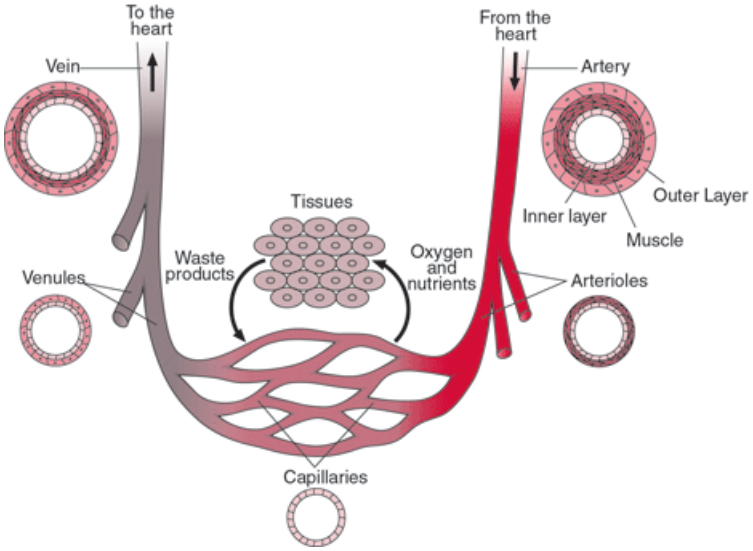


Figure 3: Different types of blood vessels[2].

mally oxygenated, exceptions made for the pulmonary and umbilical arteries. There are no valves in arteries.

2.3.2 *Veins*

In the circulatory system, veins are blood vessels that carry blood towards the heart. Most veins carry deoxygenated blood from the tissues back to the heart; exceptions are the pulmonary and umbilical veins, both of which carry oxygenated blood to the heart. Veins differ from arteries in structure and function: for example, arteries are more muscular than veins, veins are often closer to the skin and contain VALVES to help keep blood flowing toward the heart, causing them to collapse when not filled with blood. They tend to have thinner walls than arteries and their precise location is much more variable from person to person than that of arteries.

The presence of valves prevents the blood to flow back in the vessels.

2.3.3 *Capillaries*

Capillaries are the smallest of a body's blood vessels and are parts of the micro-circulation. These micro-vessels, measuring 5-10 µm in diameter, form of a network connecting the arterioles to the venules, and enable the exchange of water, oxygen, carbon dioxide, and many

2 Image available at http://www.daviddarling.info/encyclopedia/B/blood_vessel.html

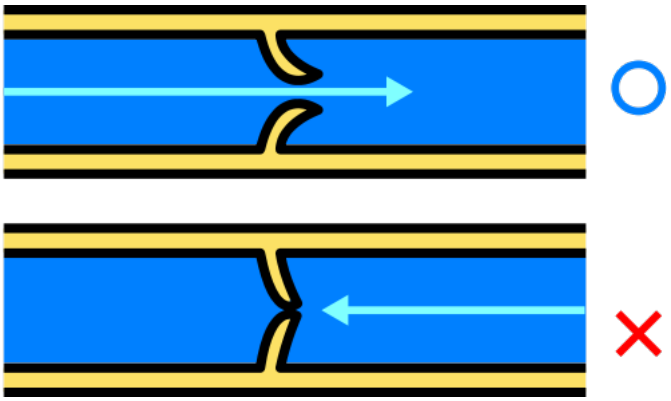


Figure 4: Functioning of valves in veins[3].

other nutrients and waste chemical substances between blood and surrounding tissues. In some areas of the body, principally the tips of the fingers and toes, there are direct connections between the arteries and veins without the intervention of capillaries. The sites of such connections are referred to as *arteriovenous anastomoses* [37].

Because of their small diameter, only large enough for RBCs to pass through in line, the blood flows slower inside capillaries than in other vessels.

2.4 BLOOD FLOW, BLOOD PRESSURE, AND RESISTANCE

BLOOD FLOW is the actual volume of blood flowing through a vessel, an organ, or the entire circulation at a given time. Blood flow through individual organs may vary at any given time.

BLOOD PRESSURE (BP) is the force per unit area exerted on the wall of a blood vessel by the blood. With blood pressure we mean the pressure of our blood in the largest arteries near the heart. Following the circulatory system from arteries to veins, we will notice that the pressure gradually decreases: this allows the blood to move, thanks to the fact that fluids move from a region of high pressure to regions of lower pressure.

RESISTANCE is the opposition to the flow of blood due to the friction between the blood and the walls of the blood vessel. Resistance to blood flow can depend upon viscosity (how thick or sticky the blood is), length of blood vessels (the longer the blood vessel the greater the resistance), diameter of blood vessels (smaller the diameter the greater the resistance).

3 Image available at http://commons.wikimedia.org/wiki/File:Venous_valve.png

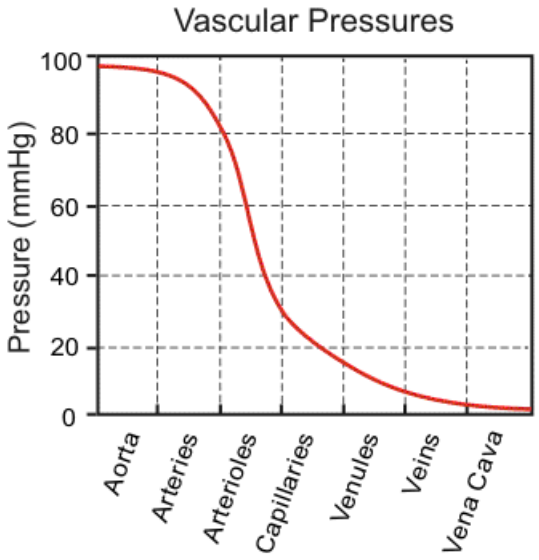


Figure 5: Pressure levels in vessels (from [26]).

From the above definition of Blood Pressure and Resistance, follows that blood flow is directly proportional to BP, and is inversely proportional to resistance.

$$\text{Blood flow} = \text{difference in blood pressure/peripheral resistance} \tag{1}$$

As shown in Figure 5, the aorta and arteries have the highest pressure. As we leave such vessels, the pressure level decreases, until reaching the lowest level in the veins, in particular around the vena cava, where is very close to zero.

Regarding the distribution of blood volume within the circulation, the greatest volume resides in the venous vasculature, where 70-80% of the blood volume is found. The relative volume of blood between the arterial and venous sides of the circulation can vary considerably depending upon total blood volume, intra-vascular pressures, and vascular compliance [26].

Elevations in BP can naturally occur during fever, exercise, and emotional upset. The following factors are believed to be involved in cases of persistent high blood pressure:

- Obesity
- Age
- Diet
- Race

- Heredity
- Stress
- Smoking

BP is measured using the blood pressure cuff (or shygmomanometer), wrapped around the brachial artery.

3 NANOROBOTS

Many words have been written about how nanorobots should be, showing an interest in nanorobotics that is growing rapidly and an active and bright community beginning to emerge. This growth of interest reflects the enormous potential of the technology and recent technical advances suggest that nanorobots will not remain in the realm of science fiction much longer.

The characteristics of a nano-agent can be easily manipulated and every modification can lead to a relevant change of the expected results. With this in mind, many researches from all over the world tried to find the best approach to various problems a swarm of nanorobots can be applied to.

However, artificial nanorobots do not exist today, primarily because of the difficulties in building the necessary nanostructures.
Nanorobotics research involves design (which often is biologically inspired), prototyping, fabrication, programming and applications such as biomedical nanotechnology.

In the next sections some of the main aspects of a nanorobot are considered, also taking into account hypothetical and futuristic solutions. We will often look towards biology, e.g. to microorganisms such as bacteria, to see how evolution has solved some of the problems nanorobots will encounter.
The aspect discussed are the following:

A. Design;

B. Intelligence and behaviour;

C. Tasks;

D. Problems that can arise.

In the last section a final design for our nanorobot will be proposed, based on what observed and discussed.

3.1 DESIGN

In this section different alternative designs for nanorobots are proposed and analysed. The purpose is to find the nanorobot design that better adapt to the environment, considering all the aspects and

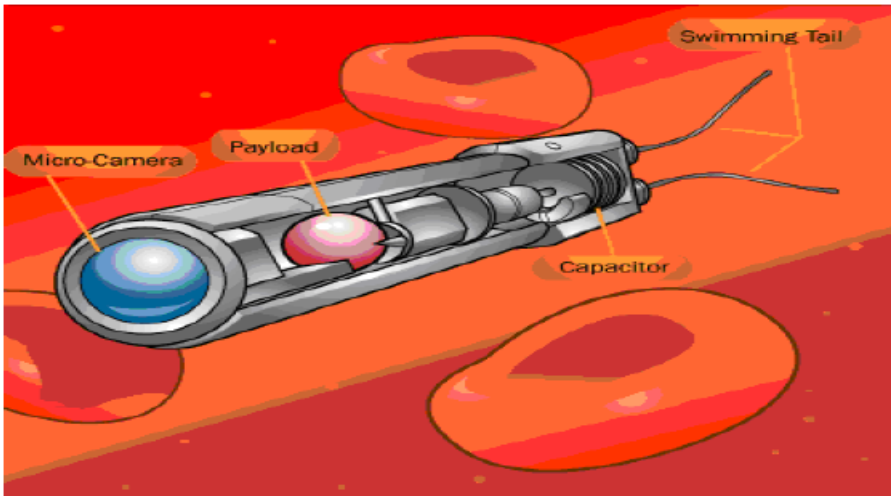


Figure 6: Concept of nanorobot using a tail to swim[1].

problems it will have to deal with.
The main aspect to consider is the robot body structure: different structures (shape, sensors, dimension) can significantly influence the tasks it can perform.

3.1.1 *Dimension*

Dimension plays a very important role in our robot definition: if the robot is too big it will not be able to explore the circulatory system in its entirety and some areas will be excluded from the study (loosing also the possibility to operate in those areas). On the other hand, if the robot is too small all the sensors and actuators it can use will be proportionally small, reducing its efficiency. If we consider to use robots for drug delivery, once again dimension plays a big role due to the amount of drug every single robot can store.

Scientists seem to agree that a robot should have an overall dimensions in the micrometer range with nanometer-scale components (typically less than 100 nm across). We will have robots with dimension close to RBCs (6-8 µm) or platelets (2-3 µm). Having a diameter that allows the robots to pass through a tight capillary, like RBCs, with the only difference that blood cells can be deformed if the capillary become too tight and robots can not (the diameter of a capillary vein is around 5-10 µm). Thanks to this the swarm of nanorobots can visit, in a theoretical way, the human body in its entirety.

1 Image available at http://sciencebox12.blogspot.nl/2010/04/how-nanorobots-will-work.html

3.1.2 *Motion system*

The first important consideration to make concerns the safety of the patient: nanorobots must be able to move around without causing damage to the host but, with so tiny dimensions, controlling where they go becomes a serious problem.

Small objects in a fluid at room temperature are subject to thermal agitation (motion) and collisions. The result is a random walk, or *diffusion*: when things get really small, lets say $1/1,000^{th}$ the width of a hair, they vibrate almost uncontrollably. It is called Brownian motion. All of this rattling around makes it hard to keep little machines in one piece and so these nanorobots would probably shake apart.

The distance L travelled by a set of diffusing objects in time t is given (approximately) by Equation 2.

$$L = (2 \cdot D \cdot t)^{1/2} \tag{2}$$

where D is the so-called diffusion coefficient, which is approximately constant for a given type of objects in a given fluid and at a fixed temperature [2]. Attempting to propel and steer a smaller organism is ineffective because of the numerous collisions that will change its course unpredictably. Diffusion is then a better strategy. It follows that self-propelled nanorobots moving in a fluid should have dimensions on the order of a few microns. Luckily, this is precisely the size one would expect to achieve by assembling a relatively complex set of nanoscale components [44].

Also it is unlikely for the robot to go against the blood flow, because of the elevated number of collisions with other particles resulting in a performance and potential loss for every nanorobot (and battery charge if battery-powered). From the results obtained by M. Zimmer during his Master Thesis research [52], we know that the particles composing blood tend to cluster in the center of the vessel (due to the bloodstream's velocity profile), following the direction of the flow. To be able to avoid collisions, the nanorobots should navigate as close as possible to the vessel walls, where such particles concentration is lower. Of course such possibility depends on robot and vessel sizes.

So far we talked about navigation and its relation with nanorobot's dimension, without taking into account the needed power: when something is very small, it is really tough to move, especially in liquids. To move along and complete the assigned tasks, the robots need a lot of power, with the result that they would need a battery about 1000 times bigger than themselves. It is very important to find a com-

promise between these aspects.

Scientists have tried to find a solution to each one of those problems, sometimes with good results.

Engine proposals

First of all, the engine problem: develop an engine that allows the robots to swim in the blood flow means building a nano-scale fully-functional energy-efficient motion system. Not a piece of cake.

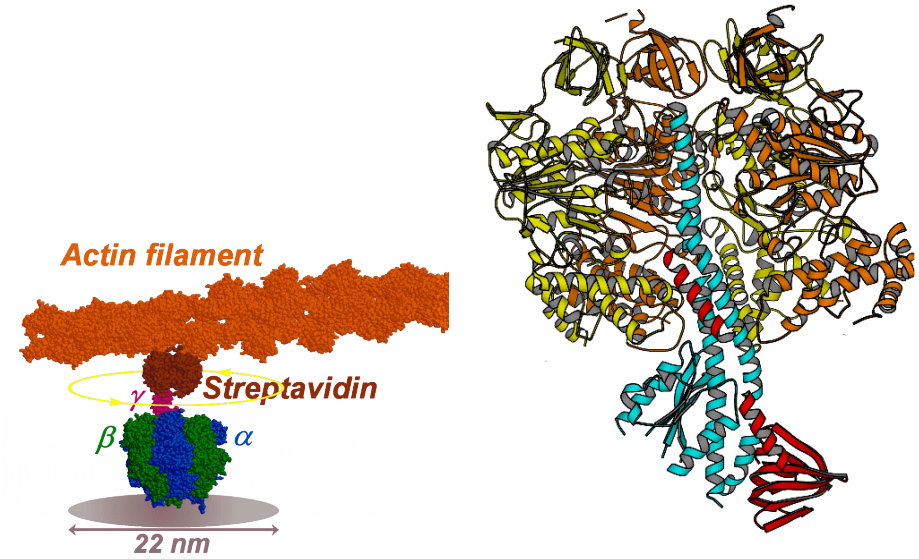
Some interesting solution have been developed:

1. USING NATURAL BLOOD FLOW
   This is the simplest way for nanorobots to explore the body, as they are just carried around by the blood flux with no need of engine. In this scenario the main problem occurs when the robots need to reach a particular point or element around them (a cell, the wall of a vessel, etc.)

2. PIEZOELECTRIC ULTRASONIC RESONANT MOTOR
   A recent developed engine that provides controllable and powerful motion at a scale appropriate for navigation in the human body (but still too big for the discussed nanorobots design). With this solution a micro-engine exploits the ability of piezoelectric materials (such as crystals) to expand or contract in response to application of a voltage, which the developing team applied a spiral structure that allows the device to develop rotational movements [50]. It is based on the same principle used by the bacteria that use a flagellum to move, like the Paramecium, making it able to swim in any direction (Figure 6).

3. SCREW DRIVES
   The propulsion is provided by two counter-rotating screw drives, enabling motion with six degree of freedom. In this model the robots can also use a navigational system to support their positioning and kinematics prediction equations [7].

4. BIOMOTORS
   Living cells, too, have engines, such as those that wave bacterial cilia or transport energy across membranes. With his team, Noji H. has investigated this molecules trying to adapt their behaviour to a solution of the engine problem [35]. The outcome is quite interesting and promising: the enzyme *ATPase* has shown the ability to rotate in response to electrochemical reactions, like hydrolysis, the process to convert ATP (adenosine triphosphate) into ADP (adenosine diphosphate). During such process, the molecule rotates in a counterclockwise direction, spinning at the rate of 3 to 4 revolutions per second. Attach a filament on



(a) ATP molecular motor (b) ATP molecule

Figure 7: Engine based on a single molecule of F1-ATPase bonded to a propeller made from protein.[2]

such "motor" was the first step to the creation of what we can classify as a biomotor (Figure 7a).

5. PUMPS
   Miniaturized jet pumps could even use blood plasma to push the nanorobot forward, moving it around like a jet airplane [22].

Engine issues

Despite some of the proposed solutions for a motor seem to be very promising, especially the biological-based, there are some issues to consider:

1. Motors running on chemical fuel (like ATP), have some drawbacks (see Section 3.1.5);
2. They operate in a narrow range of environmental conditions (e.g., temperature and pH);
3. They are hard to control;
4. They are made of soft materials of limited durability;
5. The yield of an operation is usually much less than 100%. Thus, for example, if we apply radiation of the appropriate wavelength to a solution containing light-driven molecular motors, only 10-50% actually move. Design of mechanical systems with such a high tolerance for failure is very uncommon [44];

2 Image available at http://www.k2.phys.waseda.ac.jp/F1movies/F1Prop.htm and http://www.nig.ac.jp/section/shirakihara/shirakihara-e.html

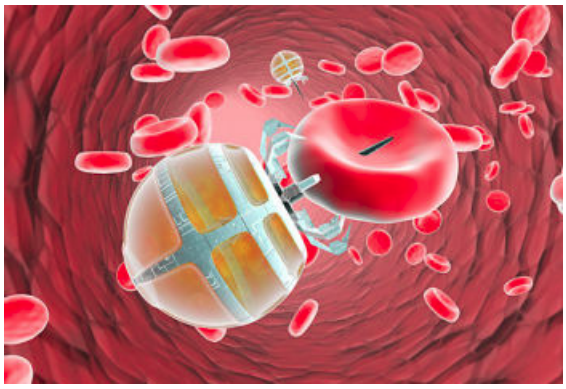


Figure 8: Spherical nanobot with actuators[3].

6. The force/torque and energy characteristics of these machines have not been investigated in detail;

7. They tend to be very complex, and much is still unknown about their structure and operation.

3.1.3 *Shape*

An important aspect to consider is the shape the robot will have. The final shape will mainly be influenced by two main factors: the environment and the function it is designed to perform. Based on this, we can divide all possible shapes in some big "families":

SPHERE This is the shape used for robots without a proper motion system (Figure 8). This type of robot moves with the blood flux and is strictly dependent on this (direction, velocity). We can think of a type of robot used to collect data from the organism and send them to an external device for patient monitoring [22]. Sensors can still be implemented.

CIGAR-SHAPED A nanorobot with a flagellum-based motion will have this particular shape, due to the fact that it has to store inside its body the engine itself and the battery to power it.

DIAMOND-LIKE STRUCTURE The nanorobot exterior shape consists of a diamondoid material (a structures that resemble diamond), to which may be attached an artificial glycocalyx surface that minimizes fibrinogen (and other blood proteins) adsorption and bioactivity, ensuring sufficient biocompatibility to avoid immune system attack [7].

3 Image available at http://www.sciencephoto.com/media/348332/view

3.1.4 *Biocompatibility*

When we think about nanorobots, we also have to keep in mind that the human body is a complex system, which has its way of regulate itself and is "programmed" to not accept unknown elements from the outside world. The introduction of an element such a nanorobot can be really problematic: we want it to be free to operate in the system, without being attacked by the immune system or the chemical substances that can be found inside the body.

Immune system response is in fact a reaction to a "foreign" surface. The problem of nanodevice biocompatibility is in principle no more difficult than the biocompatibility of medical implants generally.

Diamond coating

The main element used in the construction of the nanorobot will be carbon in the form of diamond/fullerene nanocomposites, because of the strength and chemical inertness of these forms. The best choice for the exterior coating is a passive diamond coating.

In this way it will have two surfaces: interior and exterior. The exterior will be exposed to the body fluids while the interior is a controlled environment, possibly vacuum, into which external liquids can not intrude. Passive diamond exteriors may turn out to be ideal, because it is resulted almost completely chemically inert [39].

Biopolymer incapsulation

An even more interesting solution involve the use of biometerials. Efforts to produce pure and highly biocompatible polymers have allowed scientists to apply them in several scientific areas including tissue engineering, wound dressings and drug delivery.

Alginates[4] are certainly the most frequently employed biomaterials for cell immobilization due to their abundance, easy gelling properties and apparent biocompatibility. Indeed a very high level of biocompatibility is essential assuming that the final aim of the encapsulation device is to protect the enclosed cellular tissue from the host's immune response.

However, it was recently acknowledged that the success of this therapeutic approach requires a detailed analysis, at the atomic and molecular levels, of the mechanisms involved in the biocompatibility and bioperformance of the micro-devices, as well as a step-wise approach to resolve each problem that arises [38].

We have also to define a solution for removing the robots when their task is completed: allowing them to effuse themselves via the usual human excretory channels seems to be the best solution. They

4 Alginic acid, also called algin or alginate, is a natural polymer which exists in brown seaweeds and bacteria. Its composition vary depending upon the source from which they are isolated. The production of alginates with specific structures can also be made by enzymatic modification.

can also be removed by active scavenger systems. This feature is design-dependent.

3.1.5 *Energy*

Power is a great and challenging issue. Energy can be supplied to these machines electrically, optically or chemically by feeding them with some given compound.

Nanosized battery

Between all the solutions, the most simple and plausible is to have a nano-scale battery in every robot. This option rises the problem to recharge them when the battery runs out of power, with the unpleasant consequence of replacing all the element in the swarm (sooner or later all of the batteries will finish their charge of energy). This can be a problematic solution, because replacing all the robots conflicts with the concept of a low-cost auto-sufficient swarm of units. Also the realization of such a small and efficient battery is still far from being completed.
Despite that, researchers are working on energy efficiency, which can play a major role in solving energy problem.

Recently at *MIT*[5] and *Texas Instruments*[6], a new chip design for portable electronics that can be up to 10 times more energy-efficient than present technology have been unveiled. The key to such improvement was finding ways of making the circuits on the chip work at a voltage level much lower than usual: while many current chips operate at around one volt, the new design works at just 0.3 volts [9]. With an efficient and low cost battery, the overall cost of each robot can be reduced and the replacement can no longer be so problematic.

Such small battery can be recharged using well established techniques already widely used in commercial applications of Radio Frequency Identification Device (RFID). With such device equipped, energy received can also be saved in ranges of $1\mu W$ while the nanorobot stays in inactive mode, just becoming active when signal patterns require it to do so [8].

Chemical powering

A second solution is related to the use of chemicals already in the human blood: glucose and oxygen can be used as energy source for the robots, solving the problem of energy demand (the human blood has an high concentration of those two chemical substances, which level can also be controlled by the robots, supporting the entire process). However this solution is far from being implemented, due to

5 Massachusetts Institute of Technology, also known as MIT, is a private research university located in Cambridge, Massachusetts, United States.

6 Texas Instruments Inc. is an American company based in Dallas, Texas, United States, which develops and commercializes semiconductor and computer technology.

the complexity in developing an nano-scale engine that can make use of the process and it tends to be inconvenient because it can not be easily switched on or off - a machine will move until it runs out of fuel - and normally produces waste products that must be eliminated.

Heat powering

For powering a nanorobot we could use the patient's body heat to create usable energy, but a constant gradient of temperature would be needed to manage it. Power generation would be a result of the *Seebeck effect* [13]. Since it is difficult to rely on temperature gradients within the body, it is unlikely we will see many nanorobots use body heat for power, making this solution less applicable than the previous two.

Heat is also a problem when it comes to dissipation for a nanomachine, particularly when large numbers of nanomachines are deployed *in vivo*.

3.1.6 *Sensors and actuators*

Each nanorobot would have its own elaboration unit and sensors to receive messages, detect obstacles and compute and implement the appropriate response. The type and number of sensors is strictly related to the type of work the nanorobot is designed to perform: since the space in the nanorobot is very limited, designers have to choose wisely the sensors and actuators the robot will be equipped with.

We can subdivide the type of sensors in two main categories: obligatory and optional.

Obligatory sensors

Obligatory sensors are strictly related to the task to execute and each robot in the swarm must equip them. Generally speaking, they are all those tools without which the swarm is unable to operate and the robot contribution is null. Under this group are normally found all the sensors that collect data on the environment and those actuators involved in the given task: syringes for injections, mechanical arms for manipulation of the environment, communication devices and so on.

Optional sensors

Optional sensors are, obviously, what is not strictly related to the task, but can be useful for the study and healing process. Under this category a lot of devices can be found, first of all a small miniature camera (assuming the nanorobot is not tethered or designed to float passively through the bloodstream). It is easy to see that a numerous group of nanorobots can collect in this way a lot of useful data of the environment, helping comprehending the problem and building a detailed picture of the human body. Thanks to this, a personalized

cure can be developed for the patient, leading to what the medicine of the future will be.

Some devices can also be installed on the patient's body, instead of being equipped inside each robot. Having such external system permits to assist the navigation inside the body, with stations keeping navigational coordinates and providing high positional accuracy to all passing nanorobots that interrogate them. They could be programmed to seek the areas that need support and reach them very rapidly. However the use of this type of tools should be avoided or limited, as will force the patient to deal with uncomfortable equipment and limited freedom of movement.

Actuators

For what concerns the equipped actuators, they will follow the same rules discussed for sensors: they will be strictly related to the task the robot is supposed to accomplish. However there are a few main families of actuators that are well suited for the jobs nanorobots are developed for:

- Probes, knives and chisels
  To remove blockages and plaque, a nanorobot will need something to grab and break down material
- Microwave emitters and ultrasonic signal generators
  To destroy cancerous cells, doctors need methods that will kill a cell without rupturing it
- Electrodes
  Two electrodes protruding from the nanorobot could kill cancer cells by generating an electric current, heating the cell up until it dies
- Lasers
  Tiny, powerful lasers could burn away harmful material like arterial plaque, cancerous cells or blood clots. The lasers would literally vaporize the tissue.

The two biggest challenges and concerns scientists have about these small tools are making them effective and making them safe. For instance, creating a small laser powerful enough to vaporize cancerous cells is a big challenge, but designing it so that the nanorobot does not harm surrounding healthy tissue makes the task even more difficult.

A nanorobot's target is believed to release chemical trails allowing the nanorobots to detect and recognize it. Manufacturing better nanoscale sized sensors and actuators will greatly improve such searching process.

3.1.7 *Communication*

As a swarm with a multitude of elements, it is appropriate for them to have a way to communicate. We are looking for a system to coordinate complex, large-scale co-operative activities which involve passing along relevant sensory, messaging, navigational and other data. It is also a desired requirement to be able to receive and transmit messages from and to external entities, always keeping in mind that communication by means of waves, be they acoustic, electrical or optical, is likely to be difficult because of the small sizes of the equipped antenna.

Many ways of communication exists, both in nature and in computer science, but not all of them are suited for this case of study. If we look at what nature does, we find that bees communicate directly by dancing; ants communicate by releasing chemicals (pheromones) that change the environment (this is called *stigmergy* in the robotics field); and bacteria also release chemicals, for example, to assess the number of similar bacteria near them.

First of all we must consider the effects our communication system will have on the patient's body: it must not have any side effects or harm to it in any way.

Radio signals

Following this concept, radio signals should be considered as last option for our communication system. Although implementing a RFID chip on the nanorobot seems to be simple and allows the tracking of information about the nanorobot position, scientists have so far not proven the real effects of long term exposure to such radio frequencies and a swarm of many small robots, constantly communication between each other, will expose the host body to high levels of such radiations. Research about this technology applied in the medical field is very active [45].

Ultrasonic signals

For directing the nanorobots we can also make use of ultrasonic signals emitted by each agent. These ultrasonic waves are detected by others robots using ultrasonic sensors, turning each robot in a small replica of a modern submarine. This system also allow an high-accuracy external device to listen to swarm communications and send signals back.

Chemical signals

Another smart way for robots to communicate is the use of chemical signals and *quorum sensing* [10]. Quorum sensing is the ability of nanorobots to communicate and co-ordinate behaviour via signalling molecules and it is also used by bacteria, following a very simple strategy. If each bacterium releases a fixed amount of a given chemical, it suffices to measure the concentration of the chemical to find how many bacteria are in a neighborhood [44].

On deployment, nanorobots that use quorum sensing and chemical signals could swim and start looking for the target with target-specific receptors. On reaching the specific target a nanorobot could release chemical signals in the environment which can be picked by other nanorobots and help them in homing to their specific target. After detecting the signal, a nanorobot estimates the concentration gradient and moves toward higher concentrations until it reaches the target. In this approach, nanorobots at the target release another chemical, which others use as an additional guiding signal to the target. With these chemical concentrations, only nanorobots passing within a few microns of the target are likely to detect the signal. As the signal strength increases in the environment, reaching the predefined threshold, the nanorobots could know that there are enough robots present in the target site and they could perform their designated task (say, destroy a tumour site or diagnose presence of certain foreign particles inside a patient). The vast majority of the communications between small objects such as cells and subcellular structures is done chemically, by using molecular recognition.

The sharing of information, communication and coordination makes nanorobots act more efficiently with less wandering and wasting of resources but, at the same time, poses interesting challenges for the design of robotic strategies.

3.2 PROGRAMMING AND COORDINATION

3.2.1 *Swarm intelligence*

The intelligence of each individual nanorobot is small when compared to the collection of nanorobots acting together to accomplish the given task. This group intelligence, called "swarm intelligence", helps the nanorobots do their task more efficiently and effectively in time and with fewer other resources. As a study on swarm intelligence, every robot implements the same hardware and follow the same set of rule.

Coordination is needed across the board - for communication, sensing, and acting - and poses a major research challenge. The scale and dynamics of nanorobotic systems precludes centralized coordination and global sharing of state. Therefore, we need coordination schemes that are inherently distributed and based on localized inputs, algorithms and outputs.

In nature we find a range of approaches to the coordination of large numbers of cells or organisms. For example, bacteria show very limited cordination behavior; ants use elaborate algorithms; and the human immune system has an extremely complex coordination and

(chemical) signalling scheme [44]. Evolution has produced biological systems that adapt and self-organize. How such concepts can be exploited in artificial systems is still a challenge for researchers, who came up with various solutions.

The three main swarm intelligence techniques including Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO) and Artificial Bee Colony (ABC), are all population-based algorithms with high performance in finding the optimal solution within feasible ones. For this study, based on [34], ABC is chosen to regulate the nanorobots movement.

Artificial Bee Colony model

In the ABC model, inspired by the intelligent foraging behaviour of honey bee swarms, the colony consists of three groups of bees: employed bees, onlookers and scouts. It is assumed that there is only one artificial employed bee for each food source. In other words, the number of employed bees in the colony is equal to the number of food sources around the hive. Employed bees go to their food source and come back to hive and dance on this area. The employed bee whose food source has been abandoned becomes a scout and starts to search for finding a new food source. Onlookers watch the dances of employed bees and choose food sources depending on dances.

Due to the different nature of this study, the algorithm given above needs to be adapted. Moreover, we have to control the behaviour of each nanorobot according to their defined states including normal, excite and final states. From now on we will define as "target site" the area where our defined target is located.

A. *Normal state nanorobot:* At initialization, all nanorobots are set to operate in normal state (status = 0). Each nanorobot moves along the vessel according to the ABC algorithm with influence from blood velocity within its current vessel and patterned noise simulating dynamics in vessel. During this phase nanorobots will search for their target and, when found, will try to accomplish its task. If the robots has a fixed amount of resources (drugs) to do it, it has also to control how it is used: if the task is completed but drugs still remains, it will go on searching for another target site without changing its status. In the case where there is no drug left, nanorobot will be at final state. However, if it finds other nanorobots and receives shared information on the environment, nanorobot will be at excite state.

B. *Excite state nanorobot:* When nanorobots have target information (status = 1), they will move toward their selected target site. They can inform other robots on the target information along the way. The information will be sent for a predefined number of times and, then, they will only receive information from other

nanorobots. When they find the target site, they proceed the same as they do in normal state.

c. *Final state nanorobot:* When nanorobots have no resources left (status = 2), they cannot complete their task anymore but still can travel through the vessel network to find new target sites and distribute this information to others until they are degraded and removed from the system.

Programming nanorobotic systems is a research area with strong connections with several emerging fields of computer science: sensor/actuator networks, distributed robotics and swarm intelligence. The study will be focused on the investigation of the swarm behaviour, especially of the emerging part of it, in other words not explicitly coded into the robots.

3.2.2 *Control*

Controllers for macroscopic robots are typically full-fledged computers. It is unlikely that the nanorobots of the near future will be able to carry inside of them the equivalent of a PC. But interesting behaviours are achievable with rather primitive control systems, which could probably be implemented at the nanoscale using emerging nanoelectronic technology.

Bacteria's sensing and motion behaviour

Bacteria provide an example of what can be done with a very simple control system. For example, *E. coli* move in a series of "runs" and "tumbles" [2]. A run is a motion in an approximate straight line. A tumble is a reorientation of the bacterium. An *E. coli* bacterium runs for a certain amount of time, then stops and tumbles, changing orientation to a *random* direction; it then runs again, and so on. *E. coli* manage to move towards higher concentrations of nutrients by using the following control scheme. Note that the tumble is always random, and the bacterium has no notion of where the nutrient is, or of which direction is best. All that it does is to bias its random walk, and this suffices to reach regions of high nutrient concentration. Randomness actually helps the bacterium move away from regions that become depleted, or from local minima of the concentration. The microorganism, in essence, executes a form of random search using only local information [44].

An help on navigation can also be provided by either contact sensors or infra-red sensors. If a combination of infra-red sensors is used, it could specify when both light sensors are in contact with some obstacle and return a binary "11" value, allowing to avoid the obstacle. The advantage is that the design of the motion behaviour does not change when different sensor types or alternate feature extraction

techniques are used since the information needed by it is the same binary vector in both cases [6].

3.3 SWARM'S TASK

After defining how a nanorobot is designed it is time to study what a swarm of nanorobots can actually do if free to operate inside the human body. Most of the uses are related to medical treatments but, due to the extreme flexibility and adaptations of the swarm, I believe many other uses will be found for this technology.

A. TREATMENT OF LOCALIZED MEDICAL PROBLEMS
A nanorobot can easily operate in sites normally inaccessible, for the treatment and/or elimination of medical problems where accumulation of undesired organic substances interferes with normal bodily functions, such as tumours, arteriosclerosis, blood clots leading to stroke, localized pocket of infections, kidney and liver stones and parasites removal

B. CURE FOR CANCER
Nanotechnology has the potential to radically increase our options for prevention, diagnosis, and treatment of cancer. Nanotechnology may also be useful for developing ways to eradicate cancer cells without harming healthy, neighbouring cells, using therapeutic agents that target specific cells and deliver their toxin in a controlled, time-released manner.

As a syringe is today used to inject medication into the patient's bloodstream, tomorrow, nanorobots could transport and deliver chemical agents directly to a target cell. They will be able to find and repair damaged organs or detect and destroy a tumour mass.

They would be able to communicate their positions, operational statuses, and the success or failure of the treatment as it progresses. They would tell you how many cancer cells they have encountered and inactivated. They would not only find cancers in their earliest stages before they can do damage or spread, but also deliver a small amount of a drug targeted directly at tumours, which would cause little or no side effects. Nanomedicine could result in non-invasive devices that can enter the body to determine glucose levels, distinguish between normal and cancerous tissue, and destroy the tumour in the initial stage itself [22]. A similar approach could be taken to enable nanorobots to deliver anti-HIV drugs. Such drug-delivery nanorobots have been termed "pharmacytes" by Freitas [18]

c. Nano-surgery
Each nanorobot, if provided with opportune actuators, can act as a small surgeon, with the possibility to operate directly in the site of the problem, reducing the risk of error and damage. Some simple operation can also be done by the swarm, as removing atherosclerotic plaques from blood vessel or seek and break kidney stones[7]. The use of nanorobots allows hospitals and medical centres to simplify their work, reducing risks and hospitalization times

d. Immune system augmentation
Medical nanodevices could augment the immune system by finding and disabling unwanted bacteria and viruses. When an invader is identified, it can be punctured, letting its contents spill out and ending its effectiveness. If the contents were known to be hazardous by themselves, then the immune machine could hold on to it long enough to dismantle it more completely

e. Arteriosclerosis prevention
Devices working in the bloodstream could nibble away at arteriosclerotic deposits, widening the affected blood vessels. This would prevent most heart attacks.

f. Drug delivery
Nanotechnology provides a wide range of new technologies for developing customized solutions that optimize the delivery of pharmaceutical products.

Depending on where the drugs will be absorbed (i.e. colon, small intestine, etc.), and whether certain natural defence mechanisms need to be passed through such as the blood-brain barrier, the transit time and delivery challenges can be greatly different. Once a drug reaches its destination, it needs to be released at an appropriate rate to be effective. If the drug is released too rapidly it might not be completely absorbed, or it might cause gastro-intestinal irritation and other side effects.

Nanotechnology can solve most of the problems in a new, efficient and safer way [3]

g. Implantable devices
Nanotechnology offers sensing technologies that provide more accurate and timely medical information for diagnosing disease, and miniature devices that can administer treatment automatically if required. Health assessment can require medical professionals, invasive procedures and extensive laboratory testing to collect data and diagnose disease.

Certain medical tests such as biopsies are subjective and can provide inconclusive or incorrect results. In a false negative re-

sult where a needle misses the tumour and then samples a normal tissue, the cancer may go untreated and can impact a patient's chances for long-term survival [3]. Genevogue Biotechnology introduced some possible applications:

- Retina Implants: Retinal implants are in development to restore vision by electrically stimulating functional neurons in the retina One approach being developed by various groups including a project at Argonne National Laboratory is an artificial retina implanted in the back of the retina. The artificial retina uses a miniature video camera
- Cochlear Implants: A new generation of smaller and more powerful cochlear implants is intended to be more precise and offer greater sound quality.

H. IMPROVE BODILY FUNCTIONS
The swarm of nanorobots can be used to assist and improve the normal body functions or to substitute missing and not-working elements, enabling the body to return to an healthy state. This way to operate can be considered one of the most important because has infinite applications: each robot can be used as a substitute for a platelet [34] or a red blood cell (which deficiency is very common in various anaemia forms)

I. CYSTIC FIBROSIS TREATMENT
Nanorobotics could be used in the future as a potential treatment for cystic fibrosis. This treatment could improve the life span of cystic fibrosis sufferers by reducing their risk of lethal lung infections and improving the function of their lungs too by increasing the area available for gas exchange. The treatment would also improve the quality of a cystic fibrosis person's life significantly, enabling them to breathe easier and have a better chance of playing sport or other recreational activities [21]

J. RESPIROCYTE
Respirocyte is a very interesting device presented by Hariharan, R. and Manohar, J. in their study [22]. It is an hollow, spherical nanomedical device 1 micron in diameter which would serve as a super efficient red blood cell, providing oxygen or carbon dioxide molecules. It will act like mini scuba tank within the blood allowing a person to hold their breath for more than an hour. It would be a moving thin celled tank with a brain and sensors. In an emergency situation they can be injected directly into the bloodstream of the endangered individual. Once the respirocytes have dispersed they begin releasing oxygen and collecting carbon dioxide. If an individual has lost access to a natural oxygen supply due to drowning, choking, or any other form of asphyxia, respirocytes can release oxygen throughout

the bloodstream until the danger has been removed. Respirocytes make it possible to breathe in oxygen-poor environments, or in situations where normal breathing is physically impossible.

Respirocytes could be employed as a long-duration perfusant to preserve living tissue, especially at low temperature, for grafts (kidney, marrow, liver and skin) and for organ transplantation. It could administer much larger quantities than a single red blood cell could. The reason for this is that it would be pressurized.It could also serve as a universal blood substitute, helping the treatment for virtually all forms of anaemia.

k. Chromallocyte
A chromallocyte is a lozenge-shaped mobile nanorobot, consisting of about four trillion atoms. Its purpose is to be a gene delivery vector superior to viruses (typically used today), offering a much greater degree of precision and control for the experimenter. Although the chromallocyte, designed by Robert Freitas, a pioneer in nanotechnology, has not yet been fabricated, it does seem feasible in the next few decades [19]

l. Respiratory Diseases
The devices could provide an effective long-term drug-free symptomatic treatment for asthma, and could assist in the treatment of hemotoxic (pit viper) and neurotoxic (coral) snake bites; hypoxia, stress polycythemia and lung disorders. Respirocytes could also be used to treat conditions of low oxygen availability to nerve tissue, as occurs in advanced atherosclerotic narrowing of arteries, strokes, diseased or injured reticular formation in the medulla oblongata (controlling autonomic respiration) [3]

m. Monitor patient nutrient concentration
A natural task for a swarm of nanorobots is the monitoring and controlling of the nutrients present in the host's body. This can really help to cure and deal with diseases as diabetes and HIV, which require a constant and accurate control of the state of the body. The monitoring of chemicals present in the patient's blood can also help to detect diseases at their early stage [30] (e.g., the amyloid-β protein deposits show changes on gradients as a symptom of Alzheimer disease). Once again information are crucial to achieve a more efficient treatment and result

n. Regenerating Tissues
Americans are spending a lot for anti-ageing medicines, in coming years nanotechnology will be assisting the new anti-ageing drive . It will be able to create nanorobots that can be injected into our body and these nanorobots will be capable of repairing damaged and old tissues

0. SUPER HUMAN POWERS
   Imaging a human being who can run 100 miles without getting tired, a man/ woman powerful enough to run as fast as 90 miles per second can no longer be just science fiction. These things can be possible with nanotechnology: nanorobots fused with quantum computers will be intelligent enough to alter chemicals in our body which can manipulate our functions to convert ourself into powerful humans [3].

3.4 OPEN ISSUES

In a study like the one described, it is very common and proved that many issues can rise. Some problems and related solutions are now presented.

Nanorobots injection

The first problem is the introduction of the nanorobots into the body and how to allow them the access to the operations site without causing too much ancillary damage. The size of the nanomachine determines the minimum size of the blood vessel it can traverse (see Section 3.1.1). Not only avoid damaging the walls of whatever blood vessel the device is in, we also do not want to block it too much, which would either cause a clot to form[7], or just slow or stop the blood flow, worsen the problem we want to cure in the first place. What this means, of course, is that the smaller the nanomachine the better. However, this must be balanced against the fact that the larger the nanomachine the more versatile and effective it can be [39].

In his study, Freitas R. has observed that the adult human body has a volume of about $100,000 cm^3$ and a blood volume of ~$5400 cm^3$, so adding a mere ~$3 cm^3$ dose of nanorobots is not particularly invasive. The nanorobots are going to do exactly what the doctor tells them to, and nothing more (barring malfunctions). Most symptoms such as fever and itching have specific biochemical causes which can also be managed, reduced, and eliminated using the appropriate injected nanorobots [17].

Due to the high development costs, this simple injection can have a high price and this leads to another problem, which may have a subordinate role in this scientific discussion, but is still important: the installation of such a swarm of nanorobots can be so expensive that it can be out of reach of most people, leading to a disparity in the healthcare system. This applies nowadays as well, but in an ideal

7 Blood clots form when there is damage to the lining of a blood vessel, either an artery or a vein. The damage may be obvious, such as a laceration, or may occur on the microscopic level. As well, blood will begin to clot if it stops moving and becomes stagnant

future society, nanomedicine will be affordable for everyone and without risks.

Evaluate results

With the problem related to the cost of the cure, scientists have also to find a way to define the "quality of service" given by the swarm, which can be the most important aspect in the diffusion of nanomedicine.

The main task the swarm has to accomplish will lead to a measure of quality of the service provided. This measure has to be reliable, consistent and valid for all the patients (of course every human being is different, and the swarm will have to adapt and be adapted to those differences). Having a fast and stable response is a first step to monitor the entire swarm and collect consistent and useful data of the patient. This can be achieved by having some of the nanorobots maintain positions near the vessel wall instead of floating throughout the flow in the vessel, acting as a supervisor monitoring the concentration of a signal from others; in such a way it can estimate the number of nanorobots at the target.

It will use this information to determine when enough agents are at the target site, thereby terminating any additional "attractant" signal nanorobots may be releasing. The amount is considered enough when the target region is densely covered by nanorobots, thus these tiny machines work at the target site accurately and precisely to that extent only to which it is designed to do. It is found that the nanorobots stop attracting others once enough nanorobots have responded [33].

In this way we can also handle the problem of having too many robots in the same target site, with the danger of obstruct the vessel they are working in: this is a very serious problem to avoid, because we want to not harms patient's health. And this is the fist statement we decided to follow.

Maintenance

The maintenance of the nanorobots has also to be considered an important issue: what to do with the robots that have run out of drugs or energy? The simplest solution will be to just remove them from the host body, programming them to move to particular area of the body from where being expelled (e.g. through digestion, respiratory system). This is the only way to do it and we must guarantee that the composition of the robots is not toxic or causing damage to the organism. This paradigm applies also in case of malfunction: robots should be able to be expelled from the host without damaging it.

One possible solution includes the use of a blood centrifuge (like the one used in blood donations) to separate the "dead" nanorobots from the blood itself.

Nanodoctors of the 21st century will want to remove their therapeutic nanorobots from the patient's body as soon as the nanodevices have finished the job. So there will be little danger of "old" nanorobots breaking down or malfunctioning, or causing something unpleasant to happen to the patient after the original disease or traumatic condition has been treated.

Additionally, nanorobots will be designed with a high level of redundancy to ensure fail-operational and fail-safe performance, further reducing the medical risk.

Replication

A primary information to always know is the number of robots still active in the body because, based on that, different strategies can be implemented. This information is indeed mandatory for a good realization of the project.

It is also crucial that medical nanorobots not ever replicate. In fact, it is unlikely that the FDA[8] (or its future equivalent) would ever approve for general use a medical nanodevice that was capable of *in vivo* replication. Except in the most unusual of circumstances, you would never want anything that could replicate itself to be turned loose inside your body.

Replication is a crucial basic capability for molecular manufacturing. But aside from the most aggressive applications, there is simply no good reason to risk manufacturing "fertile" nanorobots inside the human body, when "mule" nanorobots can be manufactured so cheaply, conveniently, and in such vast numbers outside of the human body. Replicators will almost certainly be very tightly regulated by governments everywhere [17].

Biocompatibility

As the first nanotechnology products begin moving into production, reports are just beginning to surface about the possible toxic effects of exposure to nanoparticles. As with naturally occurring toxic elements or chemicals, such as mercury, lead or asbestos, the concern is that engineered nanomaterials such as carbon buck balls and nanotubes could also have serious health impacts when inhaled, ingested or absorbed through the skin.

Most of the nanoparticles are considered to be safe. But it is worthy to notice, that it was not clarified yet, if all of them are not dangerous in terms of human health. As it was investigated, much in this matter

8 The Food and Drug Administration (FDA or USFDA) is an agency of the United States Department of Health and Human Services. The FDA is responsible for protecting and promoting public health through the regulation and supervision of food safety, tobacco products, dietary supplements, prescription and over-the-counter pharmaceutical drugs (medications), vaccines, biopharmaceuticals, blood transfusions, medical devices, electromagnetic radiation emitting devices (ERED), and veterinary products [16].

depends on the size of a nanoparticle involved [28].

However in this study we do not have to deal with nanoparticles that have to be stopped from entering the bloodstream, but the exact opposite! Our nanorobots will be inside the patient's body from the beginning and will not leave it until their job is done. With this precondition, researchers can focus on create the robots using just materials that are not harmful or rejected by the organism. The designing will be done in a controlled environment and having such boundaries entail two consequences: the number of material (biological or not) that can be used is limited but, on the other hand, the researchers know how they will react when inside the body (see Section 3.1.4).

Fault tolerance

The incompetence or negligence of medical personnel is always a potential concern. However, in the nanomedical era, as today, such occurrences should be infrequent and notorious.

Biocompatibility problems are well anticipated, and on-board computers should ensure safe and correct operation and reprogrammability of operational parameters even after the devices have been launched on their mission, especially to permit deactivation if anything goes wrong. Fail-stop protocols may be particularly appropriate in high-risk missions where large numbers of replacement nanorobots are readily available.

Therefore, the most serious problems may devolve from the inherent complexity of a trillion machines independently trying to cooperatively work on a very complex repair problem in a short period of time. One class of malfunction might involve some unexpected emergent machine-machine interaction-the kind of subtle interaction that is unlikely to have been exhaustively tested in full-up systems, in advance.

The doctor must always be able to "pull the plug" on the nanomachines. This is one of the most important design constraints, one that will probably become a strict and universal regulatory requirement for all medical nanodevices [17].

4

SWARM COMPUTATION: AN OVERVIEW

Dumb parts, properly connected into a swarm,
yield smart results[1].
Kevin Kelly[2]

4.1 ORIGIN AND CONCEPTS

Swarm intelligence, as a scientific discipline including research fields such as swarm optimization or distributed control in collective robotics, was born from biological insights about the incredible abilities of social insects to solve their everyday-life problems [5].

Swarm intelligence is a field of study very close to biology and it is possible to say that swarm intelligence is biology [4]. Observing and understanding the behaviour of simple entities like insects or animals, combined with mathematics and simulations, gives the possibility to develop working models to solve complex problems, also the ones not linked directly to biology.

To explain why this approach works it is necessary to start from a social paradigm point of view. The *Social Impact theory* created by Bibb Latan [29] defines three basic points to describe how a population behaves within itself.

- The behaviour of individuals can be explained in terms of the self-organizing properties of their social system
- Clusters of individuals develop similar beliefs
- Sub-populations diverge from one another

In other words, individuals tends to influence themselves becoming more similar, grouping into correlated regions within the population.

The social impact alone is not enough to explain the collective behaviour. If individuals are searching for solutions they can learn from the experiences of others (probably their neighbours) becoming more similar and improving the performance of the population. These interactions gives the population what is called intelligence.

1 From the book New Rules for the New Economy: 10 Radical Strategies for a Connected World , Penguin Books (1999)

2 Kevin Kelly is the founding executive editor of Wired magazine, and a former editor/publisher of the Whole Earth Catalog. He has also been a writer, photographer and conservationist.

Swarm Intelligence arises from these concepts as well as Evolutionary Computation paradigms. It can be defined as an interactive population of individuals that adapt itself to the local or global environment to reach a common goal. For instance in nature ants shares the goal to search for food, looking for the shortest path.

Surprisingly, the complexity of these collective behaviours and structures does not reflect at all the relative simplicity of the individual behaviours of an insect. Of course, insects are elaborated "machines", with the ability to modulate their behaviour on the basis of the processing of many sensory inputs. Nevertheless, as pointed out by Seeley [48], the complexity of an individual insect in terms of cognitive or communicational abilities may be high in an absolute sense, while remaining not sufficient to effectively supervise a large system and to explain the complexity of all the behaviours at the colony scale. In most cases, a single insect is not able to find by itself an efficient solution to a colony problem, while the society to which it belongs finds "as a whole" a solution very easily.

Behind this "organization without an organizer" are several hidden mechanisms which enable insect societies, whose members only deal with partial and noisy information about their environment, to cope with uncertain situations and to find solutions to complex problems [20].

But none of this comes without a reasonable trade-off. Biology makes a compromises between different goals and it is not clear how some natural mechanisms works. It means that sometimes biology fails [27]. Still, computer scientists are interested in this approach. The most intriguing aspects of swarm intelligence that computer scientist are interested in are the distribution and decentralization. In a swarm there are interactive autonomous agents that cooperate and organize themselves, dividing labour and distributing tasks.

Nowadays we have the possibility to apply different types of swarm intelligence algorithm, based on different types of swarm. One of the most famous is the Ant Colony optimization which simulate the behaviour of ants for foraging using a positive feedback named *pheromone*. Other approaches emulate bees, firefly, birds and even rivers.

Inspired by the research in the field of swarm intelligence, our goal is to apply such knowledge to the coordination of large number of nano-scale robots.

4.2 SWARM ROBOTICS

As seen in Section 4.1 there exists no centralized coordination mechanisms behind the synchronized operation of social insects, yet their system-level functioning is robust, flexible and scalable. Such properties are acknowledged to be desirable for also multi-robot systems, and can be stated as motivations for the swarm robotics approach:

- ROBUSTNESS requires that the swarm robotic system should be able to continue to operate, although at a lower performance, despite failures in the individuals, or disturbances in the environment. A number of factors can be observed in social insects behind the robustness of their operation.

  First, redundancy in the system; that is, any loss or malfunction of an individual can be compensated by another one. This makes the individuals dispensable.

  Second, decentralized coordination; that is, destroying a certain part of the system will not deter the system's operation. Coordination is an emergent property of the whole system. Third, simplicity of the individuals; that is, in comparison to a single complex system that could perform the same task, in a swarm robotic system, individuals would be simpler, making them less prone to failures.

  Fourth, multiplicity of sensing; that is, distributed sensing by large numbers of individuals can increase the total signal-to-noise ratio of the system

- FLEXIBILITY requires the swarm robotic system to have the ability to generate modularized solutions to different tasks. As nicely demonstrated by ants, in ant colonies individuals take part in tasks of very different nature such as foraging, prey retrieval and chain formation.

  During the foraging task, ants act independently searching for food in the environment; their search is partially coordinated by the pheromones laid in the environment

- SCALABILITY requires that a swarm robotic system should be able to operate under a wide range of group sizes. That is, the coordination mechanisms that ensure the operation of the swarm should be relatively undisturbed by changes in the group sizes.

Although we have presented the inspiration behind the swarm robotics approach, and described its envisioned properties as observed from natural systems, these by themselves are not sufficient to define the approach. For this purpose a definition of swarm robotics in given by Sahin [47]:

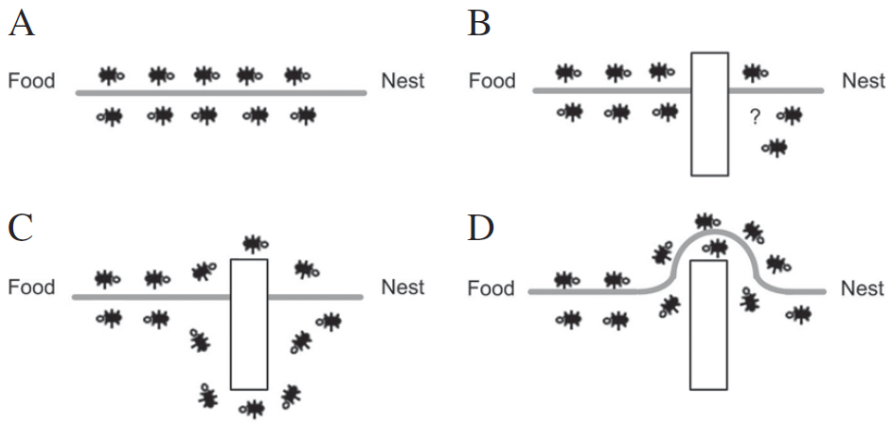


Figure 9: **A.** Ants in a pheromone trail between nest and food; **B.** an obstacle interrupts the trail; **C.** ants find two paths to go around the obstacle; **D.** a new pheromone trail is formed along the shorter path (from [40]).

Definition 1 *Swarm robotics is the study of how large number of relatively simple physically embodied agents can be designed such that a desired collective behaviour emerges from the local interactions among agents and between the agents and the environment.*

With the given definition, supported by all discussed in Chapter 3, we have now a way to understand what swarm robotic is and comprehend why it seems so popular nowadays, and not just in the science community!

4.3 MAIN ALGORITHMS

In this section the main algorithms used in swarm intelligence will be presented. For more detail about the ABC algorithm, which will be used as base for the behaviour of the simulated nanorobots, see Section 3.2.1.

- ANT COLONY OPTIMIZATION
  ACO is a class of optimization algorithms modelled on the actions of an ant colony. ACO methods are useful in problems that need to find paths to goals. Artificial "ants" -simulation agents- locate optimal solutions by moving through a parameter space representing all possible solutions. Natural ants lay down pheromones directing each other to resources while exploring their environment. The simulated "ants" similarly record their positions and the quality of their solutions, so that in later simulation iterations more ants locate better solutions [14].

  A classic example of the construction of a pheromone trail in the search for a shorter path is shown in Figure 9. In Figure 9A

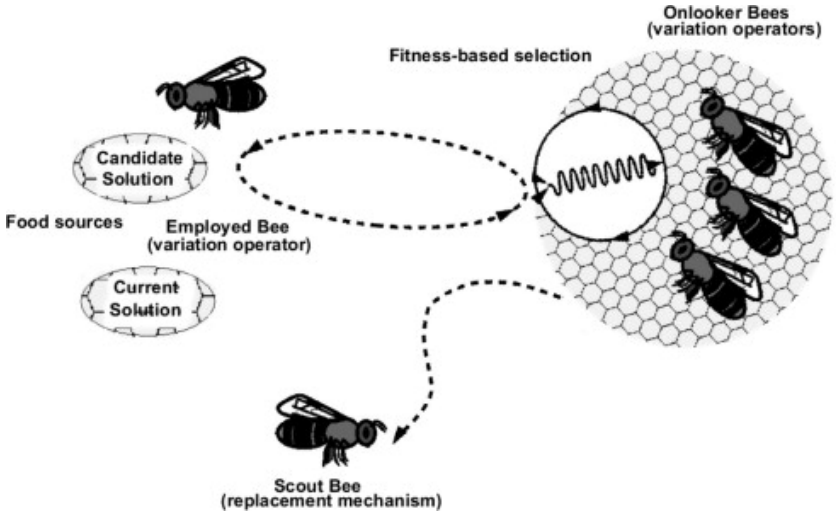


Figure 10: Graphical representation of the elements in the ABC algorithm (from [32]).

there is a path between food and nest established by the ants. In Figure 9B an obstacle is inserted in the path. Soon, ants spread to both sides of the obstacle, since there is no clear trail to follow (Figure 9C). As the ants go around the obstacle and find the previous pheromone trail again, a new pheromone trail will be formed around the obstacle. This trail will be stronger in the shortest path than in the longest path, as shown in Figure 9D.

There are still many differences between real ants and artificial ants, mainly: artificial ants have memory, they are completely blind and time is discrete. On the other hand, an ant colony system allows simulation of the behaviour of real-world ant colonies, such as: artificial ants have preference for trails with larger amounts of pheromone, shorter paths have a stronger increment in pheromone, and there is an indirect communication system between ants, the pheromone trail, to find the best path [40]

- Artificial bee colony
  ABC algorithm is a swarm based meta-heuristic algorithm introduced by Karaboga in 2005 [24], and simulates the foraging behaviour of honey bees. The ABC algorithm has three phases: employed bee, onlooker bee and scout bee (Figure 10). In the employed bee and the onlooker bee phases, bees exploit the sources by local searches in the neighbourhood of the solutions selected based on deterministic selection in the employed bee phase and the probabilistic selection in the onlooker bee phase. In the scout bee phase which is an analogy of abandoning exhausted food sources in the foraging process, solutions that are not beneficial anymore for search progress are abandoned, and

new solutions are inserted instead of them to explore new regions in the search space. The algorithm has a well-balanced exploration and exploitation ability

- PARTICLE SWARM OPTIMIZATION
  PSO is a global optimization algorithm for dealing with problems in which a best solution can be represented as a point or surface in an n-dimensional space. Hypotheses are plotted in this space and seeded with an initial velocity, as well as a communication channel between the particles. Particles then move through the solution space, and are evaluated according to some fitness criterion after each timestep. Over time, particles are accelerated towards those particles within their communication grouping which have better fitness values. The main advantage of such an approach over other global minimization strategies such as simulated annealing is that the large number of members that make up the particle swarm make the technique impressively resilient to the problem of local minima [25]

- FIREFLY ALGORITHM
  Firefly algorithm (FA) is another swarm-based algorithm, which was inspired by the flashing behaviour of fireflies. Light intensity is associated with attractiveness of a firefly, and such attraction enable the fireflies with the ability to subdivide into small groups and each subgroup swarm around the local modes. Therefore, firefly algorithm is specially suitable for multimodal optimization problems [51]. In fact, FA has been applied in continuous optimization, travelling salesman problem, clustering, image processing and feature selection.

5 A MODEL FOR THE NANOROBOT

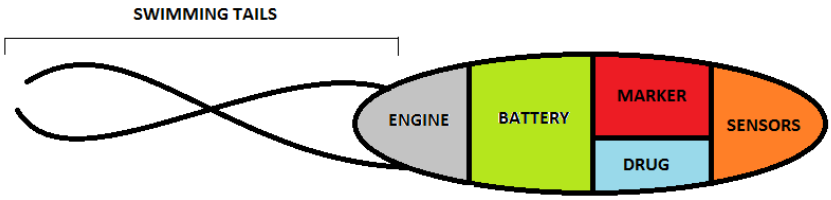


Figure 11: Concept for a future real nanorobot, based on E. coli bacterium.

5.1 INTRODUCTION

In Chapter 3 we discussed and examined the current state of the art in nanotechnology applied on human body. Many of the concepts and ideas examined are still far from being applied in a real situation but, in some years of research, the possibility of their realization exists.

Based on all the ideas and concepts in mind, a design for the implemented simulation has been chosen. All the decisions were taken considering what can be really implemented by scientists in the close future, excluding all the numerous concepts that are just theoretically applicable.

The nanorobots will follow the following design:

- Nanorobots shape will be based on the Escherichia coli bacterium, which swims in a purposeful manner, propelled by long thin helical filaments, each driven at its base by a reversible rotary engine. The robots will equip such filaments, called *flagella* and an engine to permit the motion (Figure 11)
- Dimension of a red blood cell (6-8 μm)
- Energy will be provided by a nanosized battery
- The drug needed to fulfil its task will be stored in a special compartment isolated from the outside environment
- Each robot will equip a few sensor to be able to operate or collect data of the patient's body. The number of those sensor

is deliberately kept low because of the space limitation. The equipped sensor are:
  - Chemical detector (MANDATORY)
  - Pressure sensor (MANDATORY)
  - Temperature sensor (OPTIONAL)
- Communication will be achieved using chemical signals, due to the fact that it uses the same sensors used for data collection, without the need of other networking devices
- The chemical marker used for communication will also be stored in a special isolated compartment
- A limited memory will be installed to store the rules each nanorobot will have to follow during its activity
- Last but not least, the nanorobot will equip a very simple processing unit that will give all the computation power needed.

Please note that the dimension and power of the engine, the dimension (and consequentially the capacity) of the battery and the amount of drug and chemical marker stored are strictly dependent on the dimension of the nanorobot. So, fixed the dimension, all the equipped components have to adapt to fit.

As previously said, the swarm in the simulation will try to seek and destroy cancer cells scattered inside the blood vessels, based on the chemical trace those cells release around them in the environment. To achieve this and guarantee efficient control over such a group of nanorobots with limited capabilities under highly dynamic environment, best-so-far ABC adaptation is proposed to regulate swarm behaviour. This is inspired by the robustness to changing environment of natural swarm intelligence (see Section 3.2.1).

The purpose of this chapter is to define a model for a nanorobot that can actually be realized in few years. With this goal in mind, all the option found in research papers and articles were taken into account but only what seemed realistic and close to be implemented was chosen. The realization of a simulation as close as possible to reality is what this study tries to achieve.

5.2 NANOROBOT MODEL

In this section a model for the proposed nanorobot is given. Since when an agent is designed and defined, the first step should always be the definition of a specific *task environment* (as accurate as possible), we will use the Performance, Environment, Actuators, Sensors (PEAS) description for this purpose.

5.2.1 *Performance measure*

A rational agent is one that does the right thing; conceptually speaking, every entry in the table for the agent function is filled out correctly. Obviously, doing the right thing is better than doing the wrong thing, but what does it mean to do the right thing?

The answer to this question is not so obvious but can be done in an old-fashion way: by considering the consequences of the agent's behaviour. When an agent is plunked down in an environment, it generates a sequence of actions according to the percepts it receives. This sequence of actions causes the environment to go through a sequence of states. If the sequence is desirable, then the agent has performed well. This notion of desirability is captured by a performance measure that evaluates any given sequence of environment states.

Obviously, there is not one fixed performance measure for all tasks and agents: each one should be appropriate to the circumstances.

In our case of study the performance measure will be the sequent (in order of relevance):

1. Destroy cancer cells
2. Avoid harming patient body
3. Use minimum amount of drug
4. Use minimum amount of chemical marker
5. Use minimum amount of energy

We can assign a bonus or a malus of hypothetical points at each action in relation to the satisfaction of the goals. In our list, only the first one has a bonus and all the others bring to a malus (with the difference that causing damage to the patient will have a greater malus than the others).

To study the behaviour of the swarm and the simulation itself, we need to have the ability to measure its performance in fulfilling the given task. We need a function implementing a performance measure.

This function has to consider all the given aspects of the simulation, together with terms and parameters set by the user. Speaking about elements to consider, we can distinguish them in *parameters dependent* (PD) and *parameters independent* (PI): this because a term can be set or not by the user monitoring the simulation.

Terms that follows are parameter dependent:

- l_t : life of the target the nanorobots have to eliminate

- d_i : initial amount of drug stored in the nanorobots
- m_i : initial amount of Robots Communication Marker (CoMa) the nanobots have
- n_i : initial number of nanorobots in the simulation
- e_i : initial amount of energy the nanobots have
- t_s : time units the target is set in the environment
- p_t : total number of available patches[1] a nanorobots can enter.

Parameter independent terms are:

- d_f : amount of drug present in nanorobots when the target is killed
- m_f : amount of marker stored inside the nanorobots when target is killed
- p_m : number of patches with marker on
- t_f : time units at which the simulation stops (normally because the target is eliminated)
- e_f : amount of energy the nanorobots have when target is killed.

Now we can define the performance function. To do so we have to consider a few aspects of the simulation, each one related to one of the goals stated above. The final performance function will be composed by 4 terms, to evaluate the performance related to drug usage, marker usage, time and energy consumed. We will now consider each one of them separately.

DRUG USAGE To eliminate the target the nanorobots will have to make use of the medical drug they carry. The primary goal of the simulation is of course to do so, but we have to consider that the more drug nanorobots can save, the longer they can work. Any waste of medical drug should be penalized.

The equation following will be implemented to evaluate drug usage:

$$d_f \frac{l_t}{g(d_u)} \tag{3}$$

1 The simulation's world is two dimensional and is divided up into a grid of patches. Each patch is a square piece of "ground" over which turtles can move. See Appendix A.1 for more informations about the simulation's framework.

where d_u is the drug used in the target's killing process, and is obtained simply by $d_i - d_f$. In this way we can control if the nanorobots have an amount of drug higher than the target's life, and penalize that.

The function $g(x)$, used in the formula with parameter d_u, works as follow:

$$g(x) = x + l_t \cdot \delta_x$$

where δ_{e_u} is the Kronecker delta[2], a function in two variables which is 1 if the variables are equal, 0 otherwise.

The Kronecker delta is normally expressed as show in Equation 4.

$$\delta_{ij} = \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{if } i = j \end{cases} \tag{4}$$

However, for the purpose of this study, the alternative notation δ_x is used, which is a shorter notation for δ_{x0}. The obtained Kronecker delta used in the energy's equation result as in Equation 5.

$$\delta_x = \begin{cases} 0 & \text{if } x \neq 0 \\ 1 & \text{if } x = 0 \end{cases} \tag{5}$$

MARKER USAGE When considering the usage of the marker chemical we can think in a similar way we did for the drug usage: the less we use, the better it is. However in this case there is still another thing to consider: we would like to have less markers as possible left in the environment after the target is eliminated.

We will penalize such situation but, at the same time, give a bonus for the amount of marker left.

$$m_f(1 - \frac{p_m}{p_t}) \tag{6}$$

TIME USAGE The time used to fulfil is measured in *ticks*[3] and has to be consider carefully. We can consider the mere used time, and subtract it from the calculations: in this way the less time the simulation takes, the higher the final value will be.

However we would like to consider also another aspect related to the time evaluation: the dimension of the environment. Such aspect

2 For more info about the Kronecker delta, see http://mathworld.wolfram.com/KroneckerDelta.html and http://en.wikipedia.org/wiki/Kronecker_delta

3 Again, see Appendix A.1 for more details.

can play a fundamental role in the nanorobots performance, especially if they are not so numerous. It is clear that in a very large environment populated by a few agents, finding and destroying a target can be really difficult and time consuming!

The term in the performance function related to time will be as follows:

$$t_k - \log(\frac{t_k}{\gamma}) \tag{7}$$

where t_k is the time to kill the target (obtained by $t_f - t_i$) and γ is the ratio between the number of nanorobot in the simulation and the available patches, i.e. $\gamma = \frac{n_t}{p_t}$.

ENERGY CONSUMPTION Since each nanorobot equips a battery with limited capacity, saving energy is the most important aspect of the behaviour it should have but, as in the time consuming evaluation, the dimension and number of nanorobots should also considered in the final equation: if there is just a small number of nanorobots, they will have to travel presumably for more time and distance to reach the active site, consuming more energy.

The equation for energy consumption is:

$$e_u - (1 - \delta_{e_u}) \log(\frac{e_u}{\gamma}) \tag{8}$$

where e_u is the used energy, obtained by $e_i - e_f$ and γ is the same used for time usage calculation. δ_{e_u} is the same Kronecker delta used in the drug usage evaluation (Equation 5), taking used energy e_u as parameter.

PERFORMANCE FUNCTION After defining all the single terms, we can now give the performance function that will evaluate the nanorobots behaviour in the simulation.
Combining all the pieces we will have:

$$f_o : d_f \frac{l_t}{g(d_u)} + m_f(1 - \frac{p_m}{p_t}) - (t_k - \log(\frac{t_k}{\gamma})) - (e_u - (1 - \delta_{e_u}) \log(\frac{e_u}{\gamma})) \tag{9}$$

Now that we have a definition of the scoring function, we have to define the range of values it can assume and scale all the resulting values to it. To do so we will make use of the 0-1 normalization, which normalizes the resulting values into the interval [0,1].

The maximum and minimum values the score can assume are known and are strictly dependant on the simulation parameters related to each run described above:

- The maximum value $f_{max}(i)$ is the value assumed in the best case scenario of the study (a direct injection of the nanorobots in the active site);
- The minimum value $f_{min}(i)$ is consequentially the value of the worst case, with the nanorobots failing to find and eliminate the target.

Having such values defined allows to calculate the 0-1 normalization α for the simulation score of the run i in the following way:

$$\alpha = \frac{score_i - f_{min}(i)}{f_{max}(i) - f_{min}(i)}$$

As can easily seen, if $score_i = f_{min}(i)$, then $\alpha = 0$. If $score_i = f_{max}(i)$, then $\alpha = 1$. Scaling the scoring function results makes easier to compare them and evaluate the overall performance of the simulation.

5.2.2 *Environment*

The environment the nanorobots will have to explore is what is called a Multi-agent System (MAS). With that name we consider a system composed of multiple interaction intelligent agents and their environment. In this study, robots are the agents and the circulatory system is the environment.

Generally speaking, environments can be organized according to various properties [46]:

- **Fully observable vs. partially observable (Accessible vs. inaccessible)**
  If an agent's sensors give it access to the complete state of the environment at each point in time, then we say that the task environment is fully observable. A task environment is effectively fully observable if the sensors detect all aspects that are relevant to the choice of action; relevance, in turn, depends on the performance measure. Fully observable environments are convenient because the agent need not maintain any internal state to keep track of the world;
- **Deterministic vs. stochastic (non-deterministic)**
  If the next state of the environment is completely determined by the current state and the action executed by the agent, then we say the environment is deterministic. Otherwise, it is stochastic;

- **Episodic vs. sequential (non-episodic)**
  In an episodic task environment, the agent's experience is divided into atomic episodes. Each episode consists of the agent perceiving and then performing a single action. Crucially, the next episode does not depend on the actions taken in previous episodes. In episodic environments, the choice of action in each episode depends only on the episode itself;
- **Static vs dynamic**
  If the environment can change while an agent is deliberating, then we say the environment is dynamic for that agent; otherwise, it is static;
- **Discrete vs continuous**
  The discrete/continuous distinction can be applied to the state of the environment, to the way time is handled, and to the percepts and actions of the agent. An environment is discrete if has a finite number of distinct states;
- **Single-Agent vs Multi-Agent**
  As the names suggest, a single agent environment is one which consists of only one agent. This means that this one agent does not have to account for other agents in the environment and can be solely concerned only with how its own actions affect the world. In a multi-agent environment on the other hand, agents need to account for the actions of other agents in a competitive or cooperative way, depending from the environment.

In our study the environment is considered to be the most complex scenario possible:

- partially observable
- stochastic
- sequential
- dynamic
- continuous
- multi-agent (obviously)

5.2.3 *Agents*

A simple agent program can be defined mathematically as an agent function which maps every possible percepts sequence to a possible action the agent can perform or to a coefficient, feedback element, function or constant that affects eventual actions:

$$f_a : P^* \rightarrow A \tag{10}$$

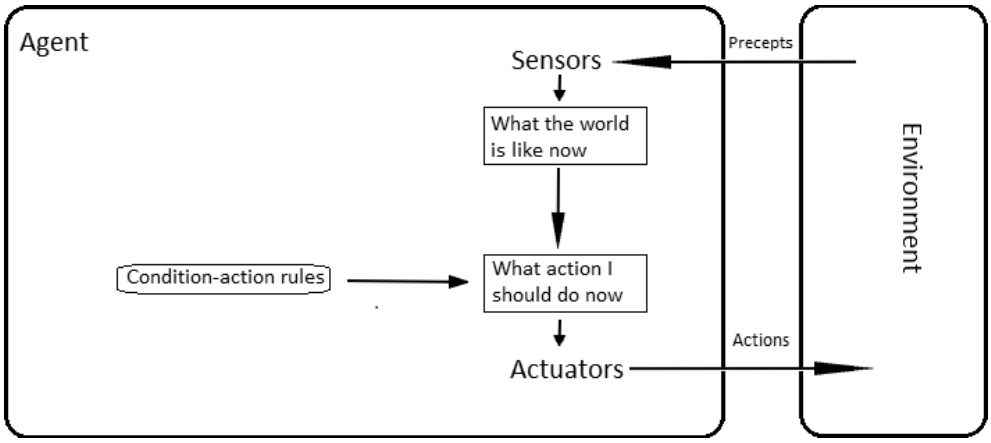


Figure 12: Schematic diagram of a simple reflex agent.

Agents in a MAS world have several important characteristics:

- **Autonomy**: the agents are at least partially autonomous
- **Local views**: no agent has a full global view of the system, or the system is too complex for an agent to make practical use of such knowledge
- **Decentralization**: there is no designated controlling agent (or the system is effectively reduced to a monolithic system)

and can be grouped into five classes based on their degree of perceived intelligence and capability [46]:

Listing 1: Pseudo code for a simple reflex agent.

```
function SIMPLE-REFLEX-AGENT(percept) returns an action
        persistent: rules, a set of condition-action rules
        state <- INTERPRET-INPUT(percept)
        rule <- RULE-MATCH(state, rules)
        action <- rule.ACTION
        return action
```

- **Simple reflex agents**
  Simple reflex agents act only on the basis of the current percept, ignoring the rest of the percept history. The agent function is based on the *condition-action rule*: if condition then action.
- **Model-based reflex agents**
  A model-based agent can handle a partially observable environment. Its current state is stored inside the agent maintaining some kind of structure which describes the part of the world which cannot be seen. This knowledge about "how the world works" is called a model of the world, hence the name "model-based agent".

- **Goal-based agents**
  Goal-based agents further expand on the capabilities of the model-based agents, by using "goal" information. Goal information describes situations that are desirable. This allows the agent a way to choose among multiple possibilities, selecting the one which reaches a goal state.

- **Utility-based agents**
  Goal-based agents only distinguish between goal states and non-goal states. It is possible to define a measure of how desirable a particular state is. This measure can be obtained through the use of a *utility function* which maps a state to a measure of the utility of the state. A more general performance measure should allow a comparison of different world states according to exactly how happy they would make the agent.

- **Learning agents**
  Learning has an advantage that it allows the agents to initially operate in unknown environments and to become more competent than its initial knowledge alone might allow. The learning element uses feedback from the "critic" on how the agent is doing and determines how the performance element should be modified to do better in the future.

We want the nanorobots to be as simple as possible, also due to the limited computational power they have, so we can think of them as simple reflex agents (Figure 12) following the instruction given as in Code 1. This implies the existence of a set of rules and actions the robot have to follow.

This statement is not a limitation because multi-agent systems can manifest self-organization and other control paradigms and related complex behaviours even when the individual strategies of all their agents are simple. Indeed, when agents can share knowledge using any agreed language, within the constraints of the system's communication protocol, the approach may lead to a common improvement.

5.2.4 *Actuators*

An actuator is the mechanism by which an agent acts upon an environment and are one of the most important aspect of a nanorobot: without them it has no real utility for the assigned tasks.
In our model the number of actuators a nanorobot can rely on is limited to two:

- Navigation engine: this is the primary method was for the nanorobot to move inside the environment, allowing to swim against the blood flow if necessary

- Drug and marker delivery system: the communications between agents are possible thanks to the markers stored inside the robots and the drug delivery system allows them to accomplish their task.

Please note that the communication to the external world can also be seen as an actuator, but it will not be included in this study.

5.2.5 *Sensors*

The number of sensors equipped should be as small as possible. This because of the small dimension of the robot and, consequentially, the small amount of available space. The equipped sensors are listed and examined in the next sub-sections.

5.2.5.1 *Pressure sensor*

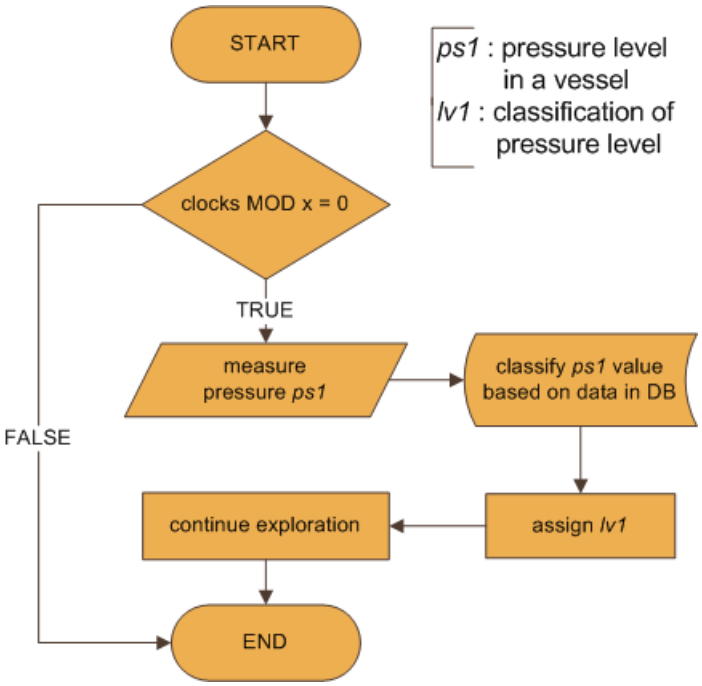


Figure 13: Pressure sensor behaviour.

This sensor is used to detect the pressure level inside the vessel. One way to do it is using the density of the blood all around the robot, but many other methods exist (e.g. piezoelectric sensor).

We want to know the pressure around the robot for a simple reason: that can help it navigate the vessels system. In the human circulatory system there are three main types of vessels: arteries, veins and capillaries. Arteries are blood vessels that carry blood away from the heart.

This blood is normally oxygenated. Veins, on the other hand, carry deoxygenated blood towards the heart, completing the circle. The only exceptions to this behaviour are the pulmonary and umbilical arteries, but in this study this particular case is not considered.

Finally, capillaries are the smallest of a body's blood vessels (5-10 μm in diameter) and are parts of the micro circulation. They are the bridge between arteries and veins and in those small vessels all the exchange of water, oxygen, carbon dioxide and nutrients takes place.

There is a constant difference in pressure between those vessels, with arteries having the highest pressure and veins the lowest. This can be used in our favour, permitting the navigation of the whole system with the definition of some rules. To do this we have to define the difference in pressure as different levels, from the highest to the lowest:

- Level 1 : Artery
- Level 2 : Capillary
- Level 3 : Vein

Please note that the pressure inside the heart is not considered (however it can considered as level 0). The way this level system is used to help robots is explained in details in Section 5.2.6.

5.2.5.2 *Chemical detector*

This sensor has to detect and analyse chemical traces and substances the robot finds during its exploration. It must be used continuously, with intervals of fixed time x: the robot must detect all the chemicals in the blood but, at the same time, using the detector too often will cause the battery to die faster. A balance between the two should be found.

The Chemical Detector is also fundamental for robots communication: based on the principle of stigmergy, a mechanism of indirect coordination between agents inspired by social insects like ants, it permits to detect traces left in the environment by other robots and decide which action perform consequentially. An indirect communication is thus created, using the detector already equipped,saving the space otherwise taken by other communication devices.

Three types of chemicals will be considered for this work

1. CoMa : this is the chemical substance a robot will release inside the vessels to communicate with other robots

2. Cancer Chemical Trace (CCT) : this is the chemical substance released by cancer cells. Robots will navigate the vessels system looking for this and, after signalling the presence of the trace to other robots, they will release the drug to destroy the cancer cell

3. Drug Trace (DT) : amount of drug present in a section of the vessel in a certain time. This is necessary to control the amount of drug released by the nanorobots.

The rules for the Chemical Detector are illustrated in the flow chart in Figure 14 and presented in the following paragraph:

- Use detector every x milliseconds (with variable x fixed)
- CCT detected
  - Distance from source $< y$: Nanorobot is close enough to the cancer cell
    - Check drug level DT in the active site
    - DT level $<$ fixed warning level β : is safe to release drug
      - Release one dose of drug (if available) or leave and continue exploration
  - Distance from source $\geqslant y$: Nanorobot is still too far from the active spot
    - Check CoMa in the current site
    - CoMa level $<$ fixed warning level δ
      - Release CoMa (if available)
      - Move toward high chemical concentration

5.2.5.3 *Temperature sensor*

This sensor measures the temperature inside patient's body. Since the only purpose of this sensor is to monitor and collect data, this is not mandatory equipment for the robots. This sensor also needs the presence of a communication system to the outside and not just the exchange of chemical markers between robots. Some basic rules can also be created:

- Measure temperature temp every x milliseconds
- When possible, send temperature data to external device

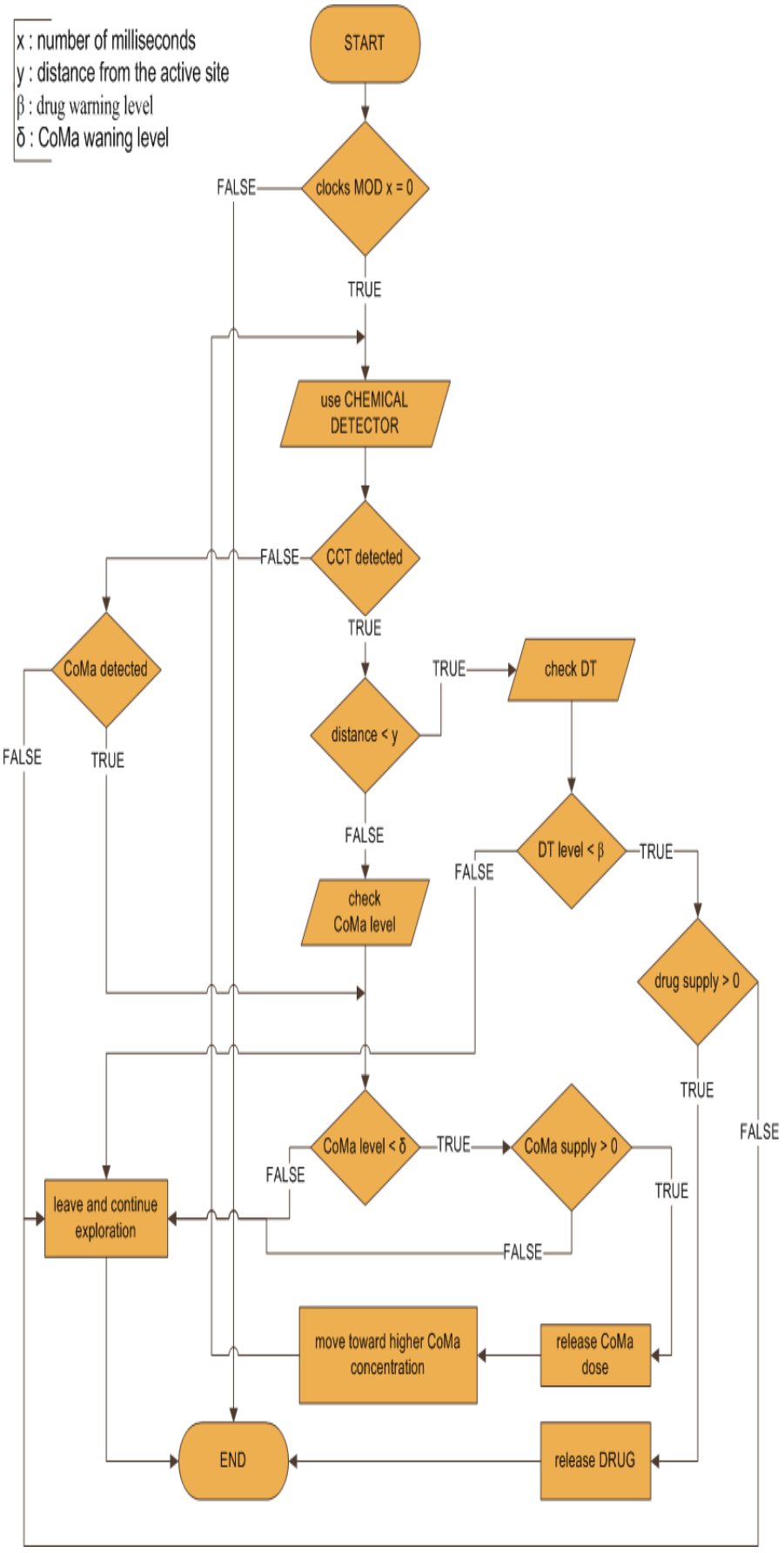


Figure 14: Chemical Detector behaviour.

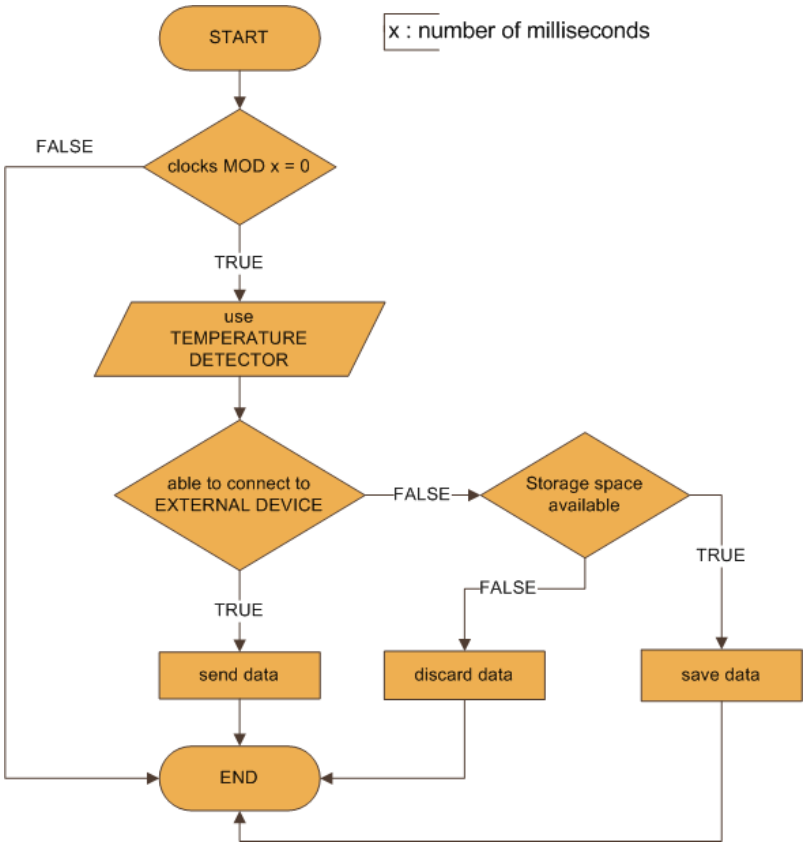


Figure 15: Temperature sensor behaviour.

5.2.6 *Navigation*

The idea is that a robot can and should find its way in the vessels system using the measured pressure levels: it can know the different levels (as fixed values) and, based on this, can choose the next vessel to enter (please see section 5.2.5.1 - pressure sensor - to more information). Robots will follow the blood stream and spread all over the body, choosing a vessel based on pressure and a probability chance (we do not want all the robots to enter a secondary artery just because the pressure is lower). Due to the human body circulatory system, the path every robot is forced to follow is the sequent: arteries, capillaries, veins and then back to arteries. No escape.

However there are situations in which arteries are directly connected to veins. This particular situation is called *arteriovenous anastomoses* and it is due to the formation of an alternative route for the blood in order to reach on organ or a tissue. *Anastomoses* normally occurs between vessel of the same type, excluding capillaries, and the direct artery-vessel connection is present only in some areas of the body, principally the tips of the fingers and toes.

Since the purpose of the simulation is to study the behaviour of the swarm of nanorobots and not to implement an exact model of the human body, the particular case of arteriovenous anastomoses will be excluded from the implementation of the circulatory system, considering the just usual blood flow.

Note that a nanorobot can always decide to go back its path to try different routes following other vessels. This is only possible in arteries and capillaries, due to the presence of valves in the veins to prevent the turning back of the bloodstream.

From all discussed, some rules follow:

- Follow main blood flow
- Measure pressure $ps1$ every x millisecond and classify $ps1$ in $lv1$ following pressure levels given - Figure 13
- Branch point is reached - Figure 16
  - Enter one of the branches (randomly)
  - Measure new pressure $ps2$ and assign level $lv2$
  - $not(lv2 = 3 \vee lv2 = 2)$: the robots is inside an artery
    - $lv2 > lv1$: changing the type of vessel
      - Follow new branch with probability $prob$ and update $ps1$ and $lv1$ or move back to branch point and select a new branch
- Pressure level $lv1 = 2$: Nanorobot is trying to enter a capillary - Figure 17
  - Reduce speed and queue with RBCs

With the rules defined a problem arises: we need the robot to be able to determine the direction it is following and change it depending on where it has to go. To do this we have to rely just on the informations the robot can obtain from the environment with its sensors. A few consideration can be done:

A. The blood flow always goes in the same direction

B. Swimming following different direction gradients (Figure 18) requires a different amount of engine power because of the blood flow

C. The robot can always know the amount of battery energy it consuming to move.

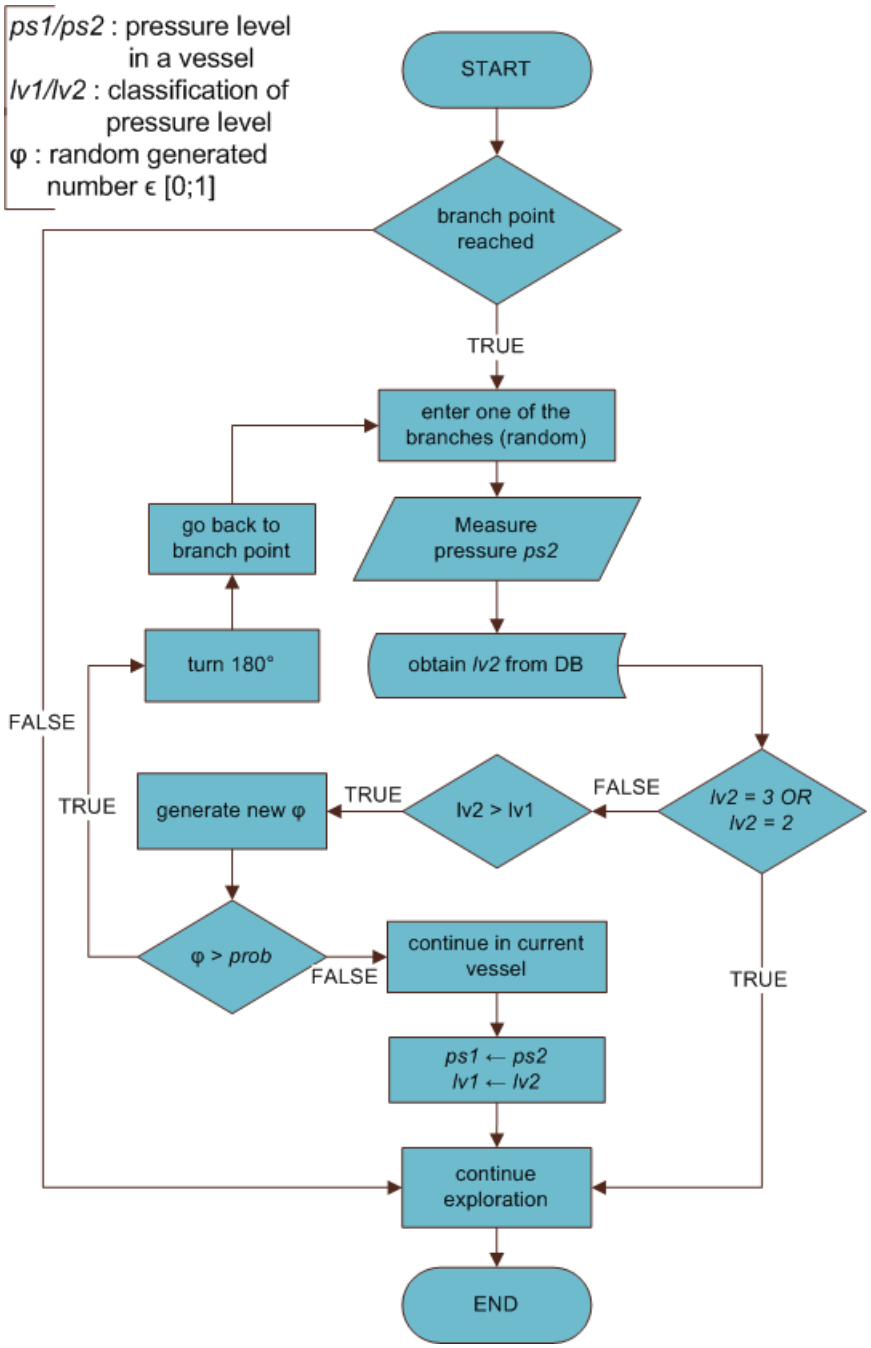


Figure 16: General navigation behaviour.

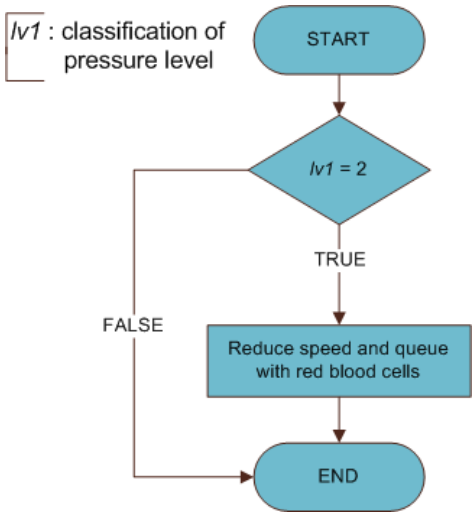


Figure 17: Entering a capillary.

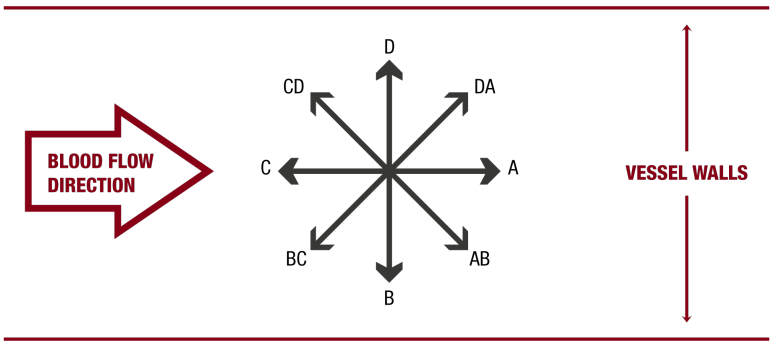


Figure 18: Possible directions a nanorobot can follow in the blood flow.

A function can now be defined, taking as input parameters the direction and the blood viscosity and giving the amount of energy used by the engine as output (Equation 11).

$$\texttt{depletedEnergy} = \texttt{f}(\texttt{direction}, \texttt{bloodViscosity}) \tag{11}$$

The next step is to subdivide the different amount of depleted energy in levels and define a conversion table from energy level to direction taken (Table 1). With he combined use of the given function and the engine power level, the robot can determine with a good accuracy which direction is following and decide when it has to stop, turn or take another path.
Therefore another simple navigation rule can be defined:

- The robot has to go back → turn 180 ° and swim back to the branch point.

Depl.Energy	Direction
1	A
2	AB
2	DA
3	B
3	D
4	CD
4	BC
5	C

Table 1: Table mapping power level to directions (based on Fig 18)

5.2.7 *Energy*

Between all the possible solution for energy supply, the simplest one was chosen: a small battery contained in the robot body, optimized to occupy the least possible space and store a high amount of energy. All the equipped sensors and the engine use the battery power.

Just a basic rule can be defined about energy

- battery level $< \alpha$
  - die (go to excretory system)

5.2.8 *General rules*

This set of rules is related to robots general behaviour. Some of them depend on the purpose the swarm will have in the body. If robots have to monitor all body functions and collect data, in addition to cure the target active sites, the general rules will change and adapt to the different task.

The rules that follow are already part of the chemical detector (see Figure 14) rule and do not consider the monitoring task:

- Drug supply = 0
  - continue exploration of the body, sending signals and collecting data
- CoMa supply = 0
  - explore the body without sending signals until battery power runs out (but still looking for active spot to cure since the robot still has some drug left)

Performance measure	Environment	Actuators	Sensors
Avoid harming patient body, Use minimum amount of drug, Use minimum amount of chemical marker, Use minimum amount of energy	Patient circulatory system: partially observable, stochastic, sequential, dynamic, continuous, multi-agent	Navigation engine, Drug and marker delivery system	Chemical, pressure, temperature

Table 2: PEAS model for the nanorobot.

5.3 PEAS MODEL

A PEAS model for our nanorobot can now be defined as in Table 2.

This model helps categorizing the task environment the nanorobots will have to deal with and also is the base on which a robot is defined. However, because of the strict hardware limitations, our implementations choices (in term of type of agent and equipped sensors/actuators) are limited but still following the given PEAS model.

Part II

IMPLEMENTATION

6 DEVELOP A SIMULATION

6.1 OVERVIEW

This chapter describes the simulation software implemented using the NetLogo framework (see A.1). All the work done is based on the model and rules described in Chapter 5, with changes depending on the framework itself. These modifications will be discussed as soon as encountered in our description.

Main features

Main features of the simulation include:

- User definition and creation of the environment;
- Possibility to save, load and modify an environment using simple text files;
- Implementation of swarm intelligence-driven agents;
- Implementation of the developed model (for both environment and nanorobots);
- Possibility for the user to manipulate simulation parameters;
- Possibility to set the target manually in the environment;
- Possibility to reset only parts of the simulation.

The system aims to be a simulation of a real-world application of nanomedicine. Because of this, between all possible solutions, only the most realistic and applicable have been chosen and implemented. A real simulation of the human circulatory system involves an high grade of complexity, mainly because the behaviour of blood in the vessel is regulated by special laws of fluid dynamic, called *hemodynamics*: the blood circulation is is pulsatile, the fluid is non-Newtonian and the vessels are of complex geometry and elasticity [41].

Since the purpose of this work is to study the behaviour of a swarm of agents, it was decided to exclude those aspects of the environment, at least in a first version of the simulation. This choice permitted to focus on the simulation itself without spending too much time of something that is important but not strictly related to the Computer Science aspect of this study.

In the next sections the system architecture, its components and their usage are presented.

Figure 19: System data flow.

6.2 SYSTEM ARCHITECTURE

The simulation has been developed with a modular design, featuring separated environment builder and simulation modules with the effort to retain efficiency, code reuse and easy system extensibility. However, since NetLogo does not implement the use of classes as data structures, the code is organized in different files, representing the different classes and two different main programs, the *Builder* and the *Simulation* , which can be considered as packages. See Section 6.3 for a more detailed description.

Data flow

The resulting data flow can be seen in Figure 19. The *Builder* has to be used first to create the test environment, representing the circulatory system model. Such model, saved on the disk as a text file, will be the input for the *Simulation* . The two software are in this way independent: each can be extended just maintaining the usual structure for the environment file. For a detailed description of the environment file structure, see section A.2.

System architecture

Since both the *Builder* and the *Simulation* have to work with the user-defined environment it is not surprising that they share some of the structures related to it. The shared structure are relative to the vessels representation and some functionalities (such as the loading function and mouse listeners) implemented in both programs. The resulting system architecture can be seen in Figure 20, where is also shown the relationship between each structure and the program that uses it.

This resulting organization permits to obtain a modular structure with the consequence that the code can easily maintained and extended.

To run properly, NetLogo requires that the two programs have access to the source code files, meaning that those files have to be stored in the same disk directory as the main programs. For what concern the environment file (which can be read or written by both programs), it can be stored in any directory, with the only requirement that the user has to have the permits to write and read files in that directory.

However keeping a dedicated directory for environment files is considered a good practice.

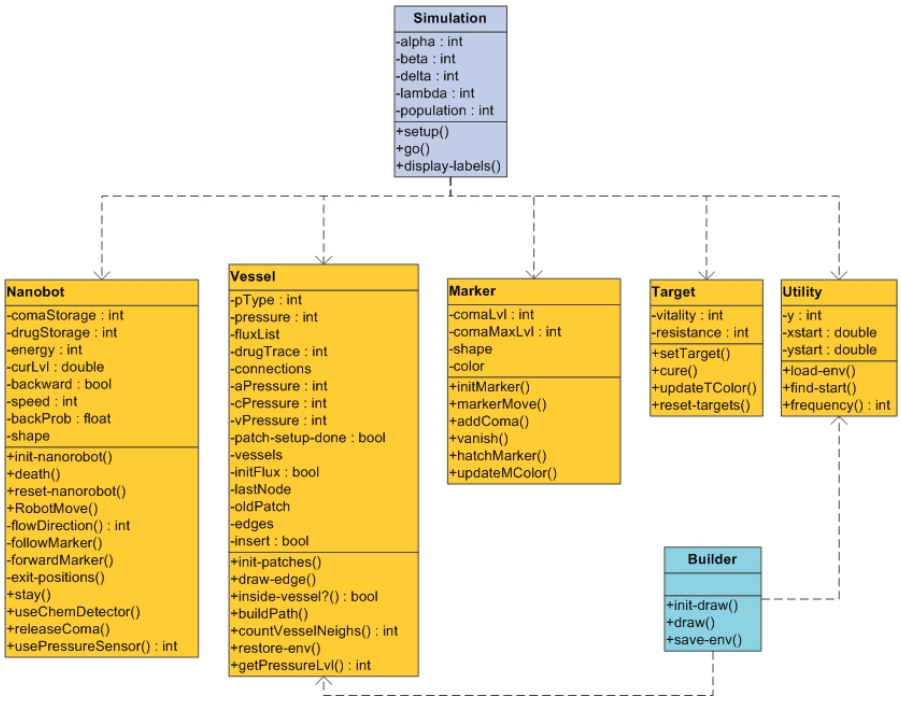


Figure 20: System architecture.

6.3 SYSTEM COMPONENTS

The simulation is based on the use of two different stand-alone software, which have to be used in a strict combination.

- BUILDER
  This is the program used to create the environment. The user has to draw the circulatory system, following some given rules. There is the possibility to save the created environment, load an existing one and modify it. For a detailed explanation see section 6.3.1;

- SIMULATION
  The simulation program, as the name says, implement the simulation itself. The user can load an existing environment and run the simulation. Many parameters can be changed to study the behaviour of the swarm. For more details see section 6.3.2.

6.3.1 *The Builder*

The *Builder* software is used to create a representation of the environment. This is achieved by directly drawing in the two dimensional grid the vessels the nanorobots will explore. This choice is due to the nature of the NetLogo framework, which forces the agents to move inside this bounded world. Through the use of such drown vessels, the user can specify a close path every agent (nanorobots, markers

and targets) can not abandon.

Creating the environment requires the user to draw all the different types of vessel and define a the direction of the bloodstream, by creating a set of edges, each connected to another forming a graph. This graph will be a directed cyclic graph $G = (V, A)$ where the set of vertices V contains all the nodes[1] and A being the set of ordered pair of vertices, called indeed edges.

The number of edges which direction points to a node v is called *indegree* and is denoted $deg^-(v)$, while the number of edges starting from node v and pointing to other nodes is called outdegree and is denoted $deg^+(v)$. In the graph the *Builder* will create the only relation between $deg^-(v)$ and $deg^+(v)$ is

$$deg^-(v) + deg^+(v) \leqslant 4 \tag{12}$$

and, for how the graph is defined, another rule follows:

$$deg^-(v) + deg^+(v) > 2 \tag{13}$$

because, as we said, every node $v \in A$ connects two or more vessels.

The only exception to Equation 13 is the Starting node from which the nanorobots begin their exploration. It has always $deg^-(v) + deg^+(v) = 2$. This graph-structure can be also seen from the environment file itself because the first section of it is the mere list of all the edges (see Section A.2 for further information).

The environment can be drawn in a grid of 50x50 patches, with the origin in the center. So the maximum coordinate for both x and y is 25 and the minimum -25. Technically speaking, since each patch measure 13 pixels on a computer screen, the grid measures 650x650 pixels. This dimension can be problematic for small resolutions but allows the user to realize more complex and extended models.

The *Builder* give also the possibility to save the work done and load existing files. To learn how to use the *Builder* to create an environment file see Section 6.4.1.

6.3.2 *The Simulation*

The simulation is implemented by the *Simulation* software. Like the *Builder* , this is a stand-alone program realized using the NetLogo framework. The aim of the simulation is the study of a swarm of simple agents, the nanorobots, involved in the task of healing a particular

1 A node is a patch which connects two or more vessels.

target. This target can be a cancer cell or a dangerous element causing problem in the patient's body.

Somebody can claim that having cancer cells floating inside a blood vessel is rarely seen and it is the symptom of an advanced state of a tumour, with such cells detaching from the original tumour and trying to survive in the bloodstream in order to find a new site to spread the illness (this secondary cancer is normally called "metastasis")[49]. This is indeed true, but what we want to study is if an efficient way to deal with problems like the one described can be found in nanomedicine.

6.3.2.1 *Simulation's agents*

The simulation involves 3 main types of agents:

A. Nanorobot
This is the most important agent and the only active one: it moves inside the environment looking for a target. It can release the CoMa and heal the target site using a special drug it can store. It has a limited amount of both CoMa and drug. Also the energy it can use is limited by the battery capacity;

B. Communication Marker (CoMa)
This agent represent the chemical signal the nanorobots use to communicate. It is normally left in one site to signal all the others to move in that direction. It has a limited existence in the environment and starts fading after a certain amount of time. Other nanorobots can add new CoMa in the same site, making it lifespan longer;

C. Target
This is a passive agent which is placed in the environment by the user and is eliminated by the nanorobots. It has a limited amount of life[2] but can implement a certain resistance to the drug used to get rid of it. The target can not move in the environment.

These 3 type of agents coexist in the simulation and all of them are present with a variable number of elements: the number of nanorobots forming the swarm can be decided by changing the related parameter in the simulation (see section 6.3.2.2), the number of marker depends by the number of robots that spots the target during their exploration and the number of targets is normally one, but nothing forbids the user to add more target agents in the simulation.

2 It is called "life" but it represents how much doses of drug the nanorobots have to use on the target to eliminate it

A particular type of agent are the patches, the squares that form the NetLogo 's world grid: those agents do not move but have their own parameter and variables which are modified by the evolving of the simulation. For a further explanation about the patches see section A.2.

6.3.2.2 *Simulation's parameters*

Many parameters are involved in the simulations, each one controlling one aspect of one type of agent. Every time the *Simulation* software is executed, the value of each parameter is set to the one used in the previous execution (their value is saved every time the simulation itself is saved by the user). This helps keeping an historical trace of the simulation's setting: if they are not changed manually by the user they will not set to different values. As expected, a variation of one parameter can greatly change the results obtained in the simulation.

The complete list of the simulation's parameters follows.

- population
  The number of nanorobots involved in the exploration. Changing its value greatly influence the results obtained: using a high number of nanorobots is considered to make the task easier. On top of that, the user has to consider the dimension of the available vessel system, in order to not overcrowd it. However its setting depends on the study the user wants to perform;

- battery-capacity
  The amount of energy each nanorobot can use. Since different type of movement like turning, stopping or going against the flow can employ different amount of energy, this parameter also effects greatly the simulation results;

- energy threshold (alpha, α)
  The amount of energy to consider a nanorobot dead. After a nanorobot reaches this level of energy it is removed from the environment (we can think it saved some energy to reach its "way out" from the environment);

- drug warning (beta, β)
  Used by the nanorobot to avoid drug overdose in the active site: if the level of **dt!** (dt!) present is higher than the fixed drug warning level, the nanorobot will not release any more dose of medical drug. Otherwise it will leave the active site and continue the exploration;

- CoMa warning (delta, δ)
  When a nanorobot sense the presence of CoMa in its current

position it has to measure its amount: if it is higher than the fixed CoMa warning level, the agent will simply move forward to its destination. Otherwise it will release another dose of CoMa, which will add to the previous one;

- target-life
  As the name suggests this is the quantity of medical drug (measured in doses) the target can endure before being eliminated. When the target reaches a target-life value equal to 0, it will be removed from the simulation.

Every parameter has a minimum and a maximum value that can not been exceeded.

To learn how to use the *Simulation* software see Section 6.4.2.

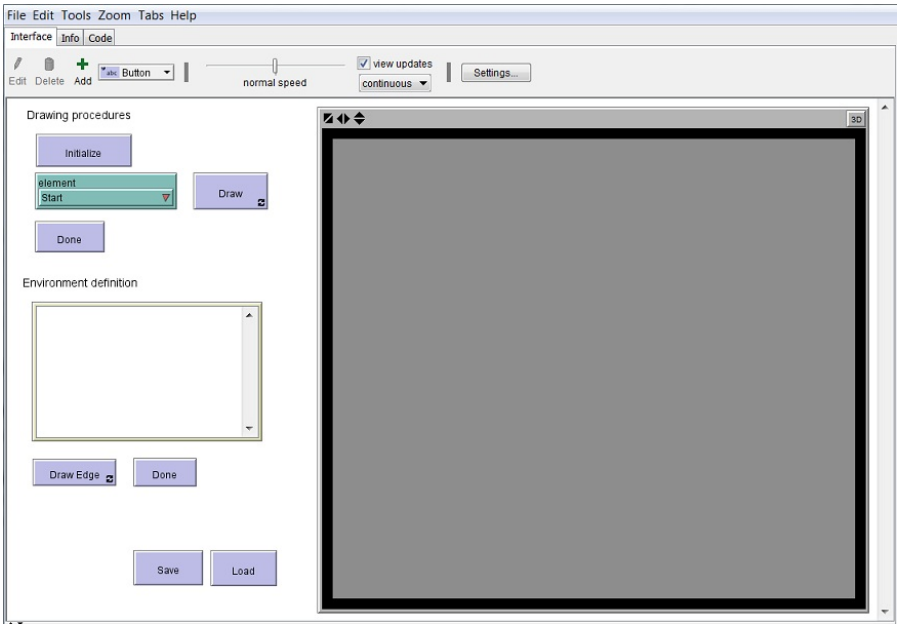


Figure 21: Builder - Initialize environment.

6.4 USING THE SYSTEM

In this section an explanation about how to use the developed system will be given. The first section explains how to use the *Builder* to create and define an environment file. The second part is about the *Simulation* software itself.

Note that there are no dependencies or particular requirements to run both program: it is only necessary that the NetLogo framework is installed in the user's machine.

6.4.1 *Create an environment*

Create a new environment

The first step, after the software initial loading is to initialize a new environment. This can be done by pressing the **Initialize** button on the top left corner (Figure 21). All the patches will now be coloured in *grey* and initialized to the default type "*wall*" (an inaccessible area of the body).

Then an option has to be selected from the drop-down menu: this option will define what type of element the user will draw in the environment. The available options are 5:

1. ARTERY : draw an artery
2. CAPILLARY : draw a capillary
3. VEIN : draw a vein

4. START : draw the Start point

5. DELETE : delete an element from the environment

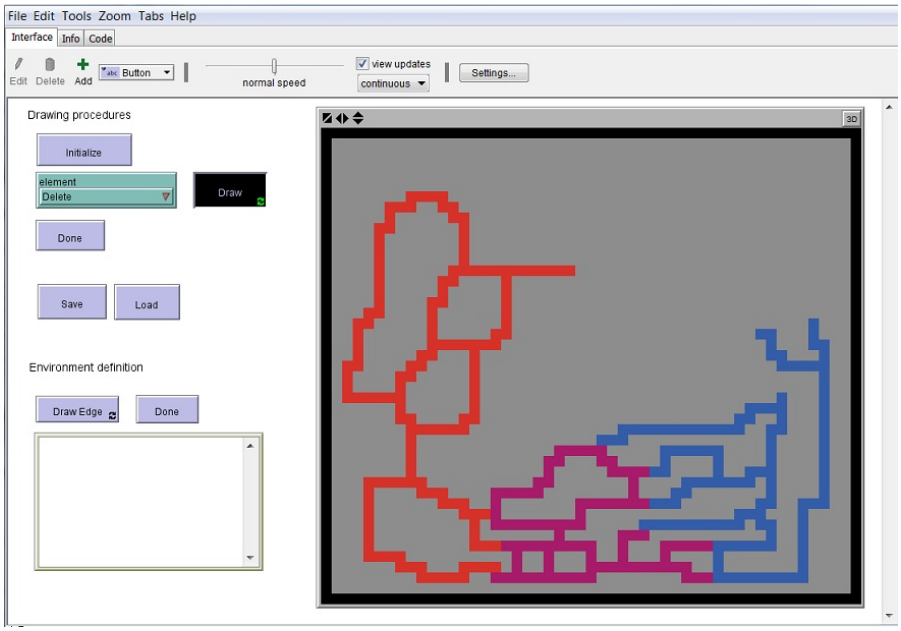


Figure 22: Builder - Drawing the environment.

As soon as one option is selected (normally one of the three available vessel types), the user can proceed with the drawing by pressing first the **Draw** button and then simply clicking inside the drawing area. Every clicked patch will turn to the color and type selected (Figure 22).

Completed the definition of the vessel system, it is very important to select and then draw one Start point. Such start point represent the location of the nanorobots injection and has to be in contact with one of the vessels. If not, the nanorobots will be stuck in such patch without any possibility of exploration. The Start point will be coloured in lime green, to make it easy to spot and different from all other patches.

After the Start point is positioned, the user has to press the **Done** button, to tell the software that the drawing procedure is completed. The software will then show a summary of all the environment informations (Figure 23).

Blood flux definition

The next step is the definition of the blood flux direction: to do so the user will have to define every edge in the environment, connecting each pair of nodes. This procedure can take some time, but is necessary to define the path the nanorobots will have to follow:

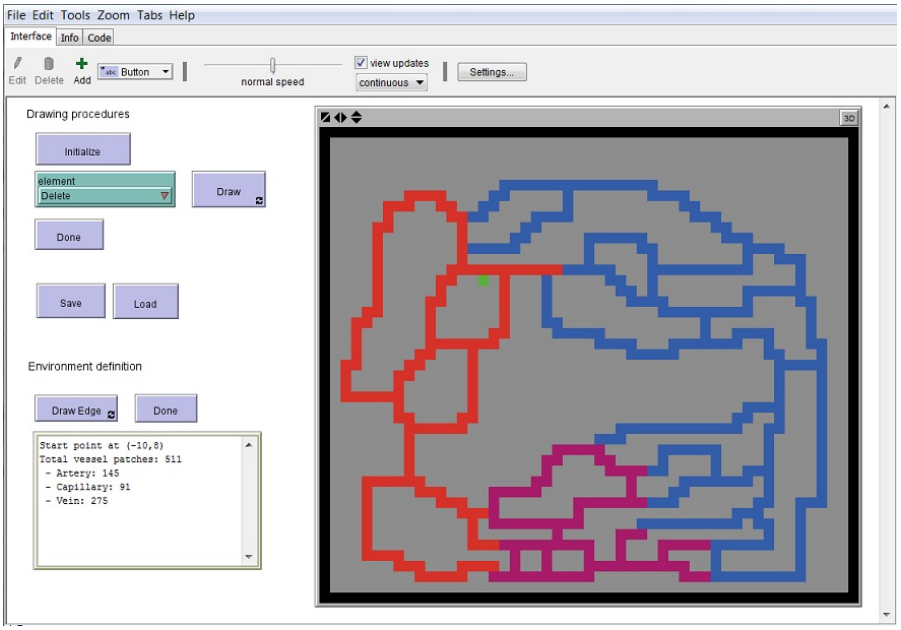


Figure 23: Builder - Finalize drawing.

without a direction of the flux, they will simply take one random direction and spread in the environment in an unrealistic way. And we want the simulation to be as close to a real world implementation as possible.

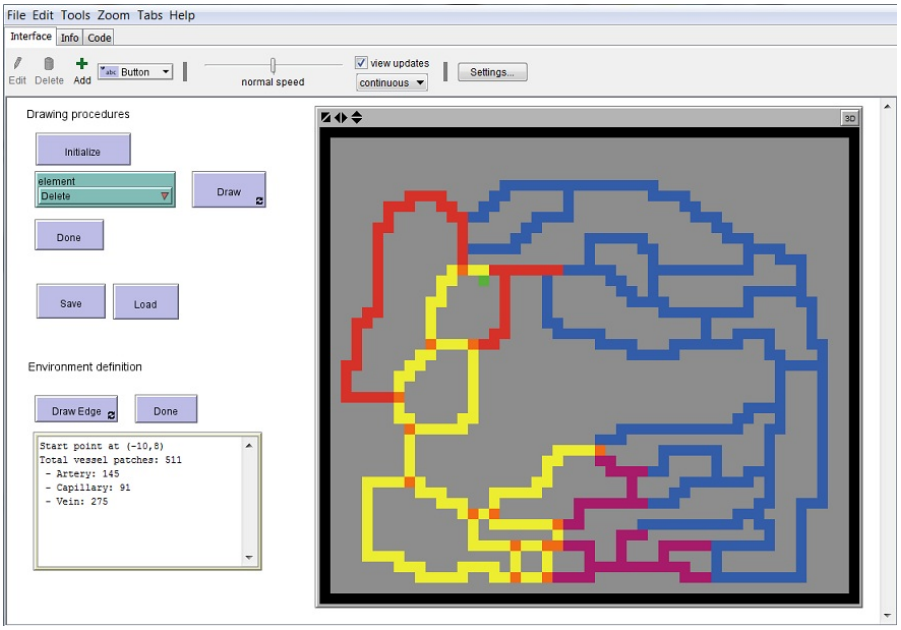


Figure 24: Builder - Edges definition.

So, what the user has to do is click on the **Draw Edge** button and click on the first node of the edge. Without releasing the mouse button, follow the vessel until another node is reached. Note that all the

visited patches will be coloured in yellow and the nodes in orange (Figure 24). Please note that is very important to start the edge definition process always from the Start point.

The same point will obviously be the final node of the last edge.

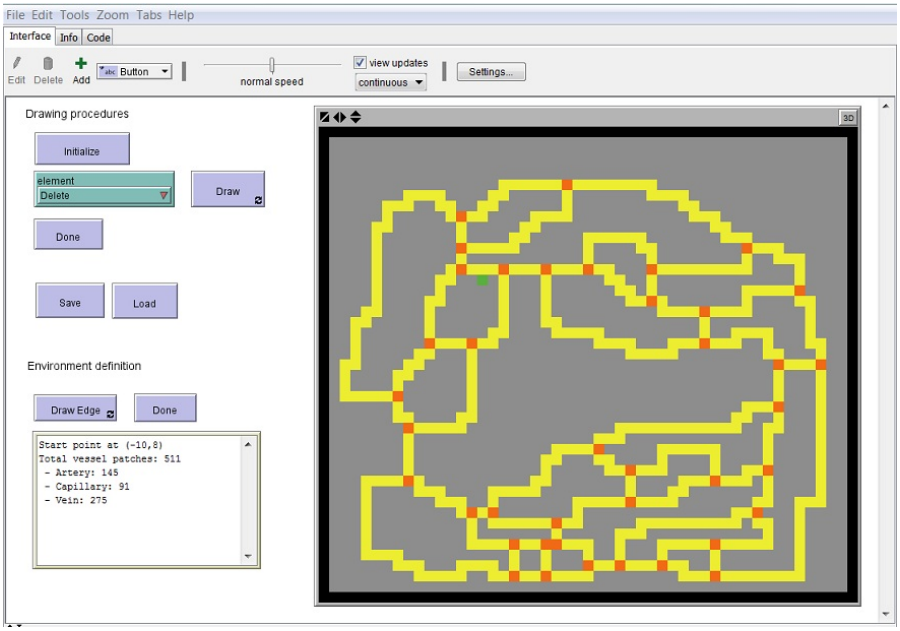


Figure 25: Builder - Finalize edges.

When all the edges have been defined and all the patches in the environment are visited, the user has to finalize this step by clicking the **Done** button on the bottom right side of the buttons area (Figure 25). All the vessels will turn to their original colours and the environment is finally ready to be used.

Save & Load

The last thing to do is to save the freshly made model, by pressing the **Save** button. A dialog window will appear and the user can select the directory to save the new environment file in.

To load an environment file, just click on the **Load** button and select the file from the dialog window that will appear[3].

3 Please note that if changes are made on a loaded files, the blood flux has to be defined again.

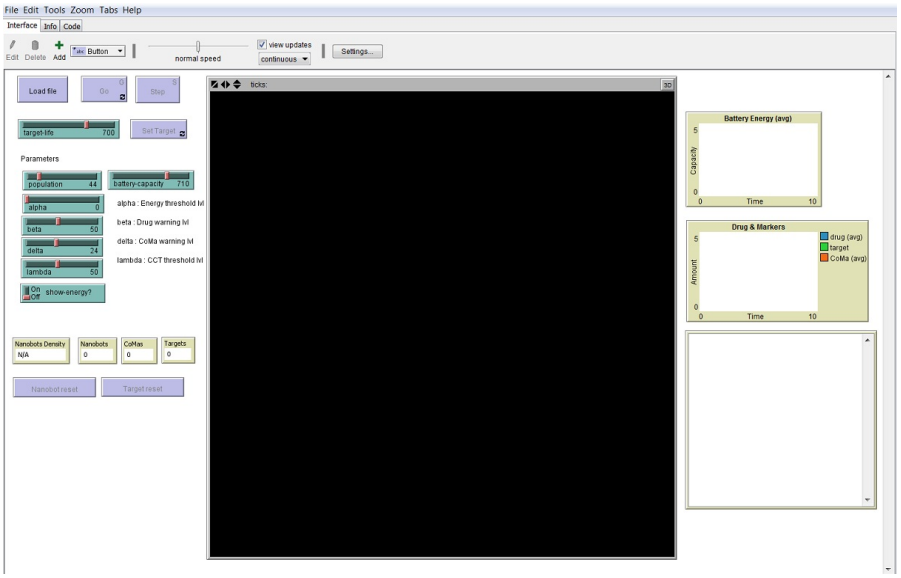


Figure 26: Simulation - New simulation.

6.4.2 *Running the simulation*

Load an environment

As soon as the program is executed, the main window will appear (Figure 26). The only available option is to load an environment file. This can be done by clicking the **Load** button on the top left corner. A dialog window will appear and the user will be able to choose the requested file from a directory on the disk.

The new environment will then be loaded in the simulation.

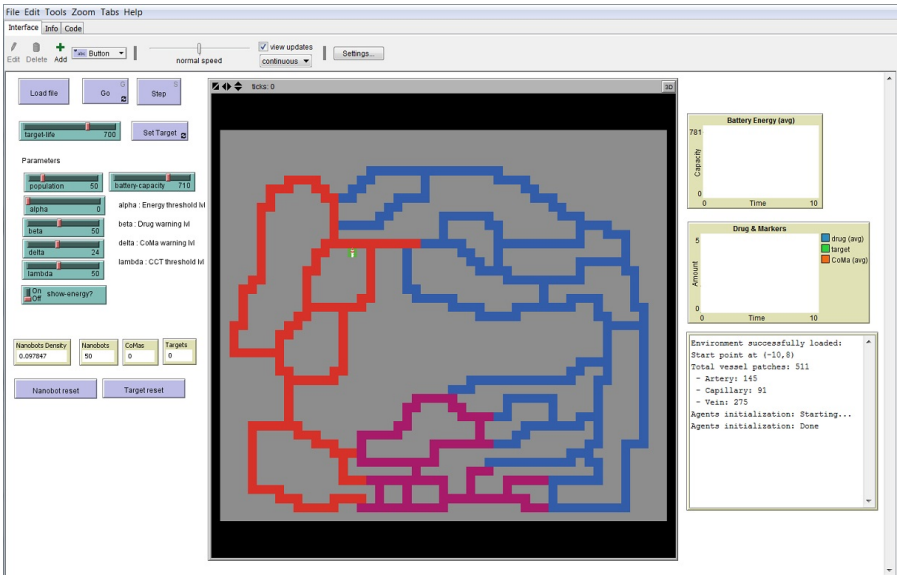


Figure 27: Simulation - Initialize simulation.

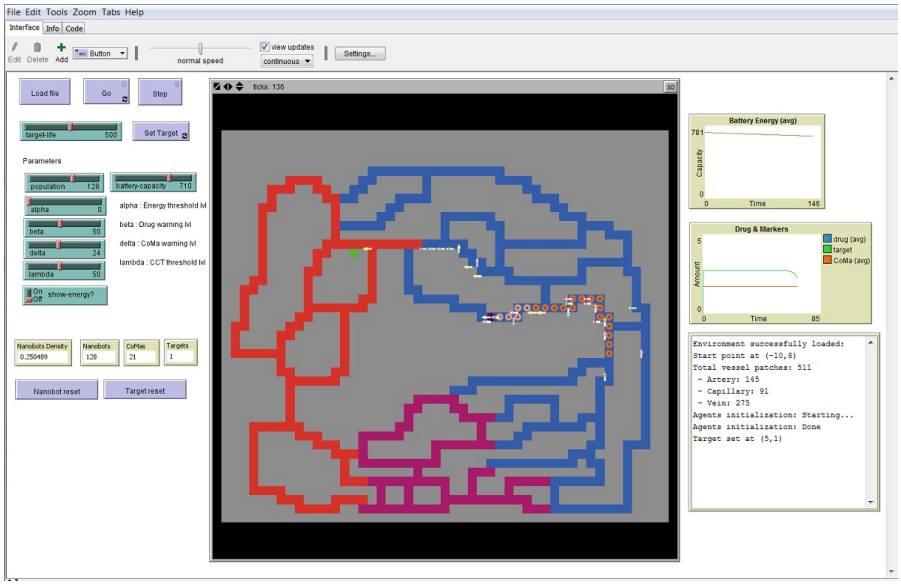


Figure 28: Simulation - Healing the target.

All the other functionalities will now become available to the user and the nanorobots will be initialized in the Starting point with the selected parameter's values (Figure 27).

Run the simulation

At this point the user is free to run the simulation by pressing one time the **Go** button, which will stay selected until all robots have died. It is possible to stop the simulation at any moment by pressing again the **Go** button.

The study the behaviour of the swarm step-by-step is also possible: instead of the **Go** button, the user has to select the **Step** button, which will advance the simulation by one step each time the button is clicked.

Add a target

Of course we want to add a target in the environment. To do so, press the **Set Target** button and click inside one of the vessel. A target will be set in that position. Note that is not possible to set the target outside a patch marked as a blood vessel. A target can be set when the simulation is in a stop state as well as it is running.

As soon as the target is set, the nanorobots will start to act as programmed, releasing CoMa and healing the target until it disappears (Figure 28). After the elimination of the target, the CoMa still inside the vessels will start to fade (Figure 29).

When there is no target to heal, the robots just navigate the circulatory system, as expected (Figure 30).

Reset the simulation

The user can also reset both the nanorobots and the targets. By pressing the **Nanorobot reset** button, all the nanorobot agents will be

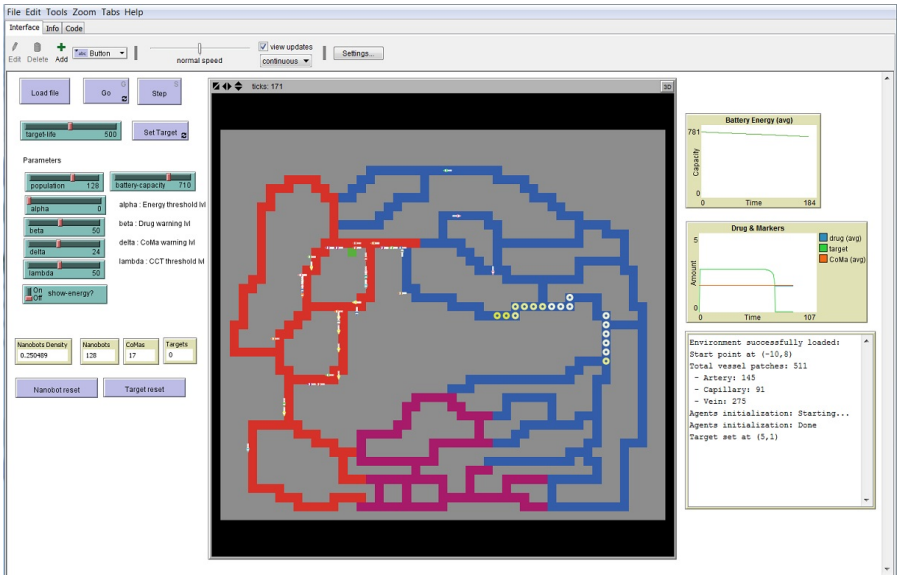


Figure 29: Simulation - CoMa fading.

placed back in the Starting point, with their energy, drug and CoMa storage back to full capacity. The simulation will be stopped.

By pressing the **Target reset** the simulation will not be stopped but all the target will just be removed from the environment.

In this way the user studying the swarm is able to control separately the two main aspects of the simulation: the swarm itself and the target to heal.

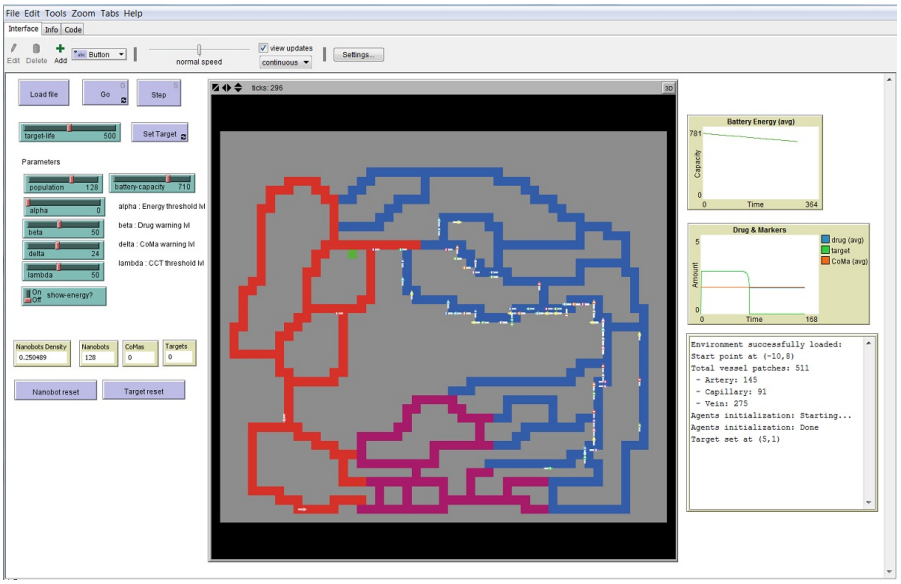


Figure 30: Simulation - Exploration.

7

EXPERIMENTS AND RESULTS

In this chapter we first introduce how the evaluation of a simulation's run is computed and proceed with presenting the obtained results.

7.1 EVALUATION

In this section an analysis of the results obtained in the simulation will be presented. First a study about the best and worst case scenario will be given, based on the performance measure formula presented in Section 5.2.1 and shown here:

$$f_o : d_f \frac{l_t}{g(d_u)} + m_f(1 - \frac{p_m}{p_t}) - (t_k - \log(\frac{t_k}{\gamma})) - (e_u - (1 - \delta_{e_u}) \log(\frac{e_u}{\gamma}))$$

7.1.1 *Best case*

The best case scenario for the study corresponds to a direct injection of the nanorobots in the active site (where the target is located). Please note that this situation is not possible in the simulation, due to the fact that setting the target directly in the starting point is not allowed. However, this case is hypothetically possible, if you think about a situation where the doctor has precise data of the patient available and a good dose of luck!

The obtained values of all the different parts of the main equation presented will be discussed in the next paragraphs.

Drug usage

DRUG USAGE The amount of drug used d_u will be equal to the target's life l_t, as expected. So, we will now have $d_u = l_t$ the given formula will change as follows:

$$d_f \frac{l_t}{g(d_u)} \longrightarrow d_f = d_i - l_t$$

where d_f in the total amount of drug left after the healing, obtained from the initial amount of drug stored in the nanorobots d_i (i.e. $d_f = d_i - l_t$) and the function $g(d_u)$ returns the value d_u received as input. Please note that when we talk about amount of drugs d, we refer to the total amount of drug present in the simulation, in other words the sum of all the single quantities each nanorobot has.

Marker usage

MARKER USAGE Since all the nanorobots in the swarm will be already in the active site, no CoMa will be released, with the consequence that there will not be patches with it. The given formula will be as follows:

$$m_f(1 - \frac{p_m}{p_t}) \longrightarrow m_f(1 - \frac{0}{p_t}) \longrightarrow m_f = m_i$$

where the amount of marker stored inside the nanorobots when target is killed is exactly equal to the initial value (again, the amount of marker is the total amount present in the environment).

Time usage

TIME USAGE The amount of time used depends on the number of nanorobots in the simulation n_t and the target's life l_t. It is measured in *ticks*, NetLogo time units. Since ticks are integer numbers, the time consumed to kill the target t_k will be calculated as $\lceil \frac{l_t}{n_t} \rceil$ and the formula will simply include this new value:

$$t_k - \log(\frac{t_k}{\gamma}) \longrightarrow \lceil \frac{l_t}{n_t} \rceil - \log(\frac{\lceil \frac{l_t}{n_t} \rceil}{\gamma})$$

Energy consumption

ENERGY CONSUMPTION Again, since the target is located where the nanorobots are, there will not be any movement and consequentially no energy consumption. Please note that in this scenario we do not consider the micro movements made by the robots to get close to the target.

The energy consumption, calculated as the total amount of energy the swarm has, will now be equal to zero in the main formula:

$$e_u - (1 - \delta_{e_u}) \log(\frac{e_u}{\gamma}) \longrightarrow 0$$

where the function δ_{e_u} is the previously seen Kronecker delta (Equation 5).

Best case

Considering the simulation's run i, the performance formula for the best case scenario, named $f_{max}(i)$, becomes as follows:

$$f_{max}(i) : (d_i - l_t) + m_i - (\lceil \frac{l_t}{n_t} \rceil - \log(\frac{\lceil \frac{l_t}{n_t} \rceil}{\gamma})) \tag{14}$$

7.1.2 *Worst case*

The worst case scenario represents the failure of the swarm in achieving its main goal: the nanorobots are not able to find and kill the target and are removed from the environment after running out of energy.

The obtained values of all the different parts will be as follows.

Drug usage

DRUG USAGE Since the swarm was not able to find the target, no healing operations were executed. This means that in this worst case no drug is used. So, we will now have $d_u = 0$ and the given formula will change as follow:

$$d_f \frac{l_t}{g(d_u)} \longrightarrow d_f \frac{l_t}{l_t} \longrightarrow d_f = d_i$$

where d_f in the total amount of unused drug, corresponding to the initial amount of drug stored in the nanorobots d_i. The function $g(d_u)$ returns the value 1, because of the received input being equal to 0.

Marker usage

MARKER USAGE In this hypothesis, resulting from the failure in finding the target, no CoMa has been released and no patches present traces of it. The given formula will be as follow:

$$m_f(1 - \frac{p_m}{p_t}) \longrightarrow m_f(1 - \frac{0}{p_t}) \longrightarrow m_f = m_i$$

where m_i is the initial given value.

Time usage

TIME USAGE The time factor needs some special considerations: at first sight it seems impossible to evaluate such aspect of the simulation, since the user can not know how much time it will take for the robots to run out of energy. Having a different energy consumption for different types of movements (Equation 11), means that the simple consideration $1 timestep = 1 movement$ is indeed wrong.

However we can still find a good estimation for the energy usage in this worst case, based on the considered scenario itself: the fact that the target is never found means that the agents will spend all their time and energy exploring the environment. This exploration phase can be considered, without losing correctness, just a sequence of steps in order to move inside the vessel following the blood flow.

Each time step an agent can execute a limited movement actions:

- HOLD ON
  this action allows the robot to maintain its position. Since the robot has to oppose the flow it requires a good quantity of energy
- MOVE FORWARD
  the robot move forward in the vessel, following the blood flow. Requires almost no energy, depending from the movement speed
- MOVE BACK
  the robots is moving against the blood flow, going back to an already visited location
- TURNING
  to turn, the robot has to oppose the flow, even if it is for a small amount of time. As for the holding on action, it requires quite some energy.

Since in this worst case the robots are exploring the environment looking for the target, the HOLD ON action will not be considered because the robots are supposed to stop just when the target is found.

To evaluate the time usage we will consider the MOVE FORWARD action: this is indeed the simplest movement a nanorobot can perform, with the least energy cost[1].

Considering the exploring phase formed by simple steps represents the case in which the nanorobots last as much as possible, corresponding to the definition of an upper bound for this aspect of the study.

From now on we will call λ the assigned cost for the MOVE FORWARD action. The time usage will be evaluated considering the amount of energy stored initially in each the nanorobots $\frac{e_i}{n_t}$ divided by λ, giving us what we can properly consider the maximum number of time steps the swarm will be active (always considering that the number of time steps is an integer number).

The time usage formula will become as follow:

$$t_k - \log(\frac{t_k}{\gamma}) \longrightarrow \lceil \frac{e_i}{n_t \lambda} \rceil - \log(\frac{\lceil \frac{e_i}{n_t \lambda} \rceil}{\gamma})$$

Energy consumption

ENERGY CONSUMPTION Since the target is not found, the agents will move around in the environment until their energy supply empties and they are removed from the simulation. This means that the energy consumption will be the total amount of energy available at the beginning e_i.

1 The energy cost for every movement is decided by the user and defined in the simulation's code

$$e_u - (1 - \delta_{e_u}) \log(\frac{e_u}{\gamma}) \longrightarrow e_u - \log(\frac{e_u}{\gamma}) = e_i - \log(\frac{e_i}{\gamma})$$

where the function δ_{e_u} returns the value 1, making the evaluation possible.

Worst case

Considering the simulation's run i, the performance formula for the worst case scenario, named $f_{min}(i)$, becomes as follows:

$$f_{min}(i) : d_i + m_i - (\lceil \frac{e_i}{n_t \lambda} \rceil - \log(\frac{\lceil \frac{e_i}{n_t \lambda} \rceil}{\gamma})) - (e_i - \log(\frac{e_i}{\gamma})) \quad (15)$$

7.1.3 *Evaluate a run*

Now that we have defined how to determinate a lower and an upper bound for the simulation, it is time to evaluate each single run.

With the term run we define the sequence of steps, beginning with the selection of the GO button in the Simulation program and ending with either the elimination of the target or the "death" of all the nanorobots.

Each run depends on the given parameters and a random factor. Due to this, is expected that two runs using the same parameters will have a different outcome. The aim of this study is to study and evaluate such results. The evaluation of each run i is based to the idea that, having a lower and an upper bounds it can assume, will be possible to assign a score to such run normalizing the obtained result in the 0-1 interval thanks to the normalization process (see Section 5.2.1).

The score of run i, named $score_i$, is initially computed using the presented performance formula in its "standard" form and, based on the parameters given in such run, the upper bound $f_{max}(i)$ and lower bound $f_{min}(i)$ are determined and used to compute the final score α.

$$\alpha = \frac{score_i - f_{min}(i)}{f_{max}(i) - f_{min}(i)}$$

Thanks to the definition of α, we can now compare different runs and start to collect simulation's data.

7.2 EXPERIMENTS

Before moving to examine the experiment's results, some words have to be spent about the test configuration and the choices made.

First, build and run an experiment in the simulations means running it with a certain parameters configuration and observe the outcome. Analysis and considerations are made based on such results.

Indeed, the defined $\alpha - score$ can give us an idea of the overall quality of the results, but an external observer is still needed.

For our test setting some conventions have been determined:

- The starting point for the nanorobots will always be placed on an artery vessel. This does not have any special reason but is done to keep this aspect constant and coherent with all the environment files tested;
- The target, represented with a purple mark, is placed in the environment in a casual point (patch) but its position is maintained for all the tests run on that environment;
- The special case in which the total amount of drug in the simulation (i.e. stored in the nanorobots) is not sufficient to heal the target will not be considered. The goal of this study is to observe how effectively the swarm fulfils its task and the way it is done. On top of that, it is expected that the doctor administering the treatment will know the amount of medicine required and injects nanorobots in a sufficient number to be able to heal the patient;
- The number of nanorobots is also a very important parameter to consider. Some experiments focus on the variation of the population of agents but for all the other experiments the ratio between environment's dimension and number of agents have been kept equal to $\gamma = \frac{1}{3}$, in other words we consider having 1 agent every 3 patches. The reason for this is that there is a maximum number of nanorobots that can be injected in the human body and having too many in our simulation's environment would not represent a real scenario of application. The chosen ratio is indeed quite high, but we have to consider also that the environment we use is very limited in dimension and using less then 10 agents means that we can not consider our group of nanoborots a swarm anymore.

The parameter analysed are those already examined during the definition of the model in Chapter 5. A brief list is given in Table 3.

The study has been done on 3 different environments, leading to 3 different subsets of experiments. Each environment present a dif-

Population	number of nanorobots in the environment
Energy	amount of energy available to each nanorobot
CoMa	amount of communication marker stored in each nanorobot
Drug	amount of medical drug stored in each nanorobot
α	threshold level for the energy
β	drug warning level in the active site
δ	CoMa warning level in a vessel
back probability	probability for the agent to turn and go back at a branch node
mFW	energy cost for a single step forward
mBack	energy cost for a step swimming against the blood flow
mStay	energy cost to be stationary inside a vessel
mFT	energy cost to move forward after a change of direction
Target life	amount of drug the swarm has to release to eliminate the target
γ	ratio between number of agent and environment's dimension
UB	upper bound for the simulation's score
LB	lower bound for the simulation's score
$\alpha - score$	score assigned to each run of the simulation
Time	time used (in *ticks*)

Table 3: Experiments parameters.

ferent configuration, with the dimension (i.e. the number of available patches) as the main distinguish aspect. Considering this, we can classify the 3 as a simple, a medium and a complex experimental environment.

This classification allows the experiments to observe on the swarm's behaviour in configurations with different levels of complexity.

In the next subsection we will examine all the selected environment in details.

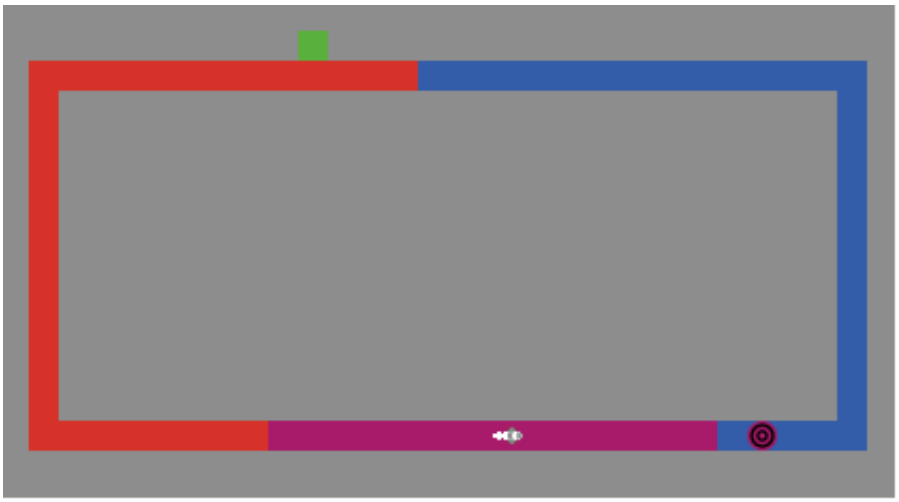


Figure 31: Experiment 1 - Simple.

7.2.1 *Simple*

The first experiment configuration is, as the name says, *simple*. The vessel system forms a loop in which the target is placed. There is no possibility for the agents to not spot it, because of the forced path they have to follow (Figure 31).

See Table 4 for the environment data and Table 5 for the initial parameters configuration.

Total patches	78
Arteries	32
Capillaries	15
Veins	31
Start point position	(-8, 12)
Target position	(7, -1)

Table 4: Simple experiment data.

This experiment helps understanding how the system works and how the score is assigned. However it is too simple to really observe an interesting behaviour of the swarm.

Population	Energy	Coma	Drug	α	β	δ
100	700	50	50	0	30	25
back prob	mFW	mBack	mStay	mFT	Target life	γ
0.3	0.2	1	0.3	0.5	500	0.1282
UB	LB					
9496.5911	-63489.8266					

Table 5: Experiment 1 initial configuration.

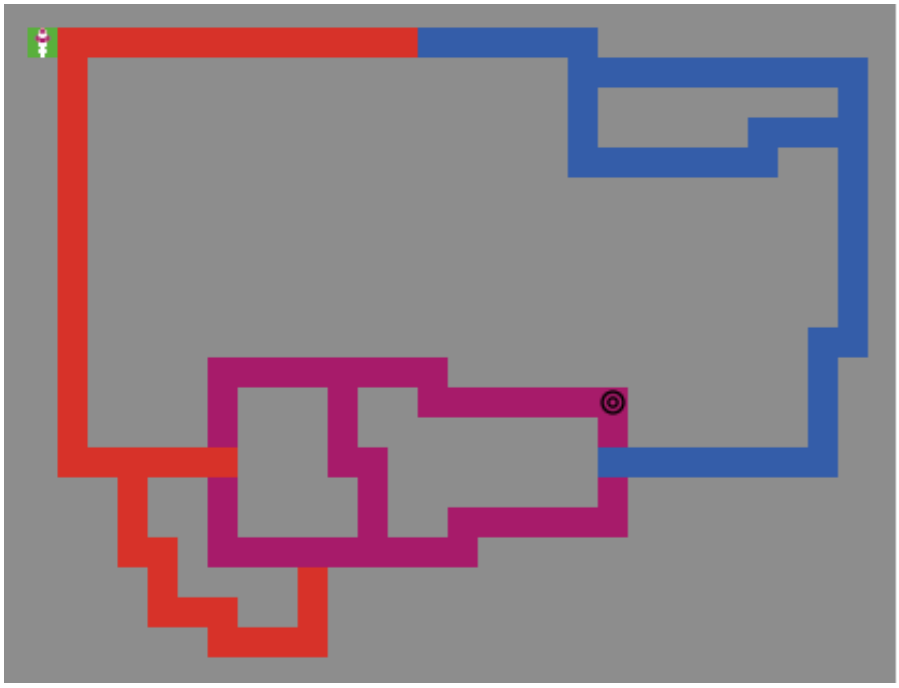


Figure 32: Experiment 2 - Medium.

7.2.2 *Medium*

In this second experiment, the environment is formed by a more complex structure, but still with the idea of having the path for the agents forming a loop. Since of the possibility for each agent to follow a different path in the system, we classify this experiment as *medium difficulty* (Figure 32). Now the possibility for the agents to not encounter the target in their navigation exists.

Table 6 shows the experiments data and Table 6 the initial parameters configuration.

Total patches	136
Arteries	45
Capillaries	42
Veins	49
Start point position	(-15, 8)
Target position	(4, -4)

Table 6: Medium experiment data.

Population	Energy	Coma	Drug	α	β	δ
100	700	50	50	0	30	25
back prob	mFW	mBack	mStay	mFT	Target life	γ
0.3	0.2	1	0.3	0.5	500	0.7353
UB	LB					
9495.8325	-63491.3438					

Table 7: Experiment 2 initial configuration.

7.2.3 *Complex*

The third experiment involves a complex vessels structure which, still following the main concept of the loop, includes numerous branch points and alternative routes (Figure 33). In this network of vessels, finding the target becomes more complicated, so cooperation between the agents is requested to fulfil the given task.

This is indeed the most interesting case of study, the closest to what a small section of the human circulatory system could be.

Initial parameters configuration and environment data are given in Table 9 and Table 8.

Total patches	536
Arteries	179
Capillaries	98
Veins	259
Start point position	(-17, 5)
Target position	(16, -9)

Table 8: Complex experiment data.

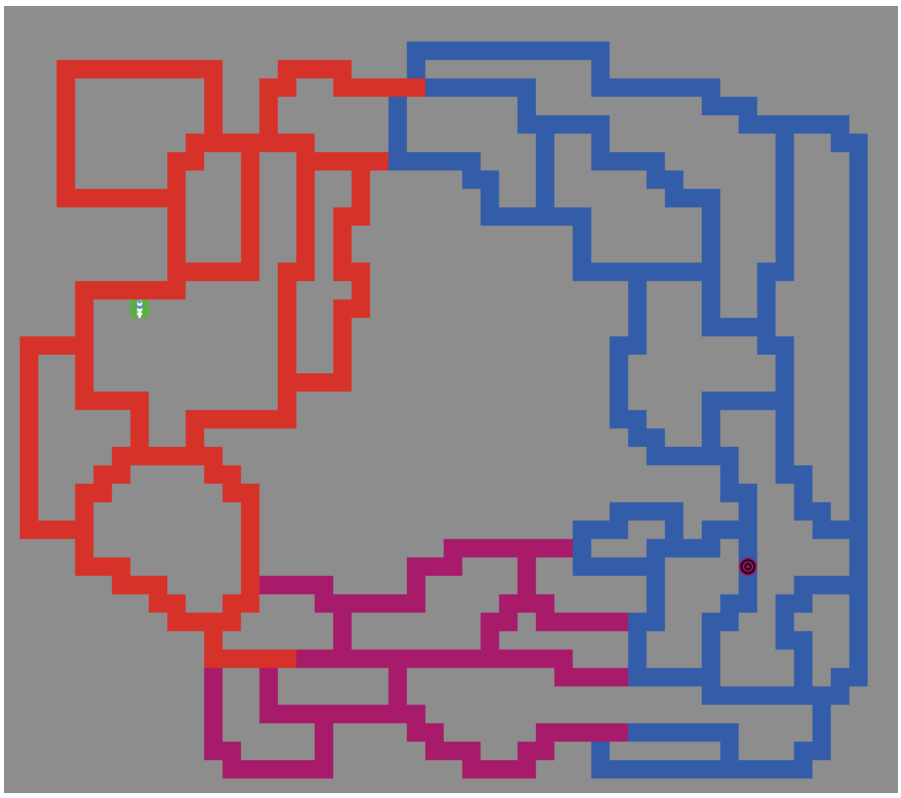


Figure 33: Experiment 3 - Complex.

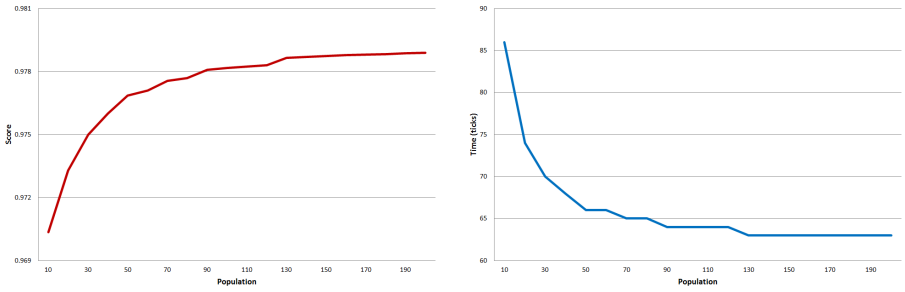
Population	Energy	Coma	Drug	α	β	δ
175	500	50	50	0	30	25
back prob	mFW	mBack	mStay	mFT	Target life	γ
0.3	0.2	1	0.3	0.5	500	0.3265
UB	LB					
16997.9633	-72490.6878					

Table 9: Experiment 3 initial configuration.

7.3 RESULTS

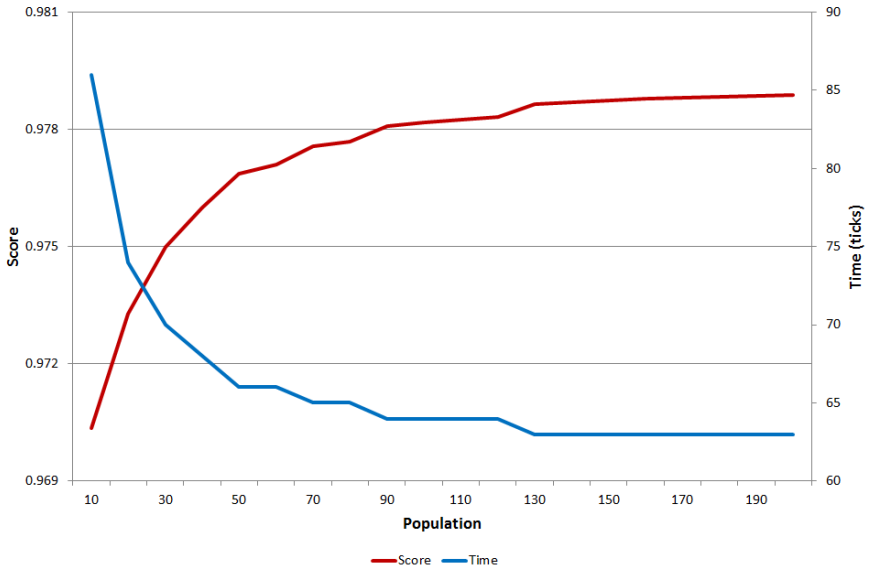
In this section results of the experiments are given and discussed.

7.3.1 *Simple*



(a) Score variation.

(b) Time variation.



(c) Combined graphs.

Figure 34: Results for population variation.

As anticipated the simple configuration is indeed...simple! The environment is a loop and it is impossible for the nanorobots to not encounter the target. With this in mind, we can proceed to take a look at the obtained results.

Population study

The first test was done to study the behaviour of the simulation with a variation of the population of agents. The number of nanorobots has been modified from the minimum of 10 to a maximum of 200, obtaining the results visible in Figure 34. As expected, as the number of agents increases, the $\alpha - score$ increases as well and the time consumption decreases. This is a direct consequence of the fact that

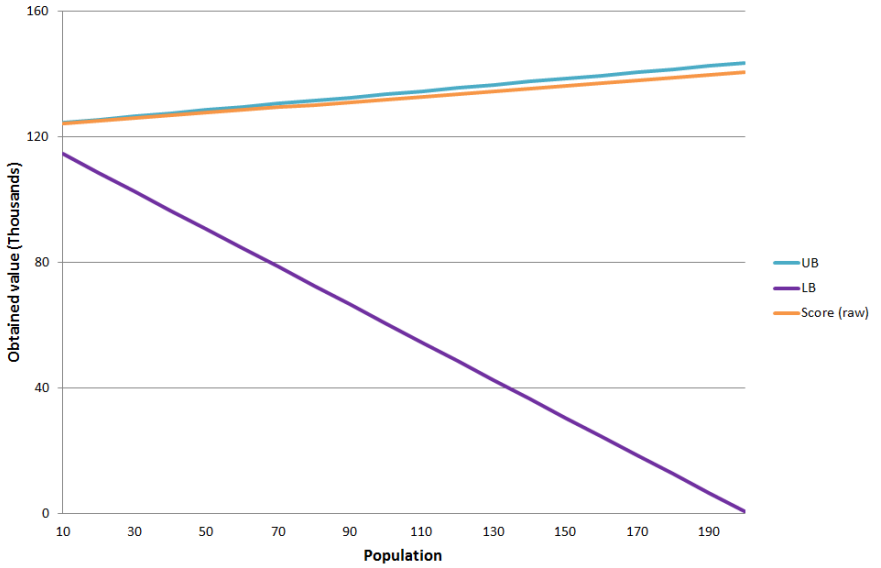


Figure 35: UB, LB and raw score comparison with population variation.

having more agents involved in the healing process makes it faster (remember that at each time step a nanorobot releases a dose of healing drug).

Also, after reaching a certain amount of agents, the trend seems to stabilize, due to the fact that, in an environment this simple, exceeding a certain threshold for their number will not increase the performance: the simulation will simply have a lot of nanorobots not participating in the healing process because of the high number of agents already fulfilling that task.

In this particular experiment this threshold can be considered approximately the value of $\gamma = 1.7$, strengthening the belief that an high number of agents does not improve the quality of the results, does not make a good use of resources (many agents do not work at all) and should be avoided.

If we observe the Upper bound (UB) and the Lower bound (LB) compared to the obtained score values (Figure 35) we can easily see that the score is always very close to the UB, mainly because of the simplicity of the experimental setting. The values presented in the graph have been translated by the function $f(x) = x + 14,000$ (i.e. moved by $14,000$ units), to avoid them to be negative.

UB and LB, as expected increase (or decrease) linearly with the population's dimension. The reason is quite simple, as the formulas computing both bounds is strictly related to that parameter.

Target study

A second test involves the variation of the target's life l_t: with a fixed population's value of 100, which makes γ even lower than the

discussed $\frac{1}{3}$, we proceed to progressively increment the l_t value, starting from 300 and ending with 1000. This has lead to the results show in Figure 36.

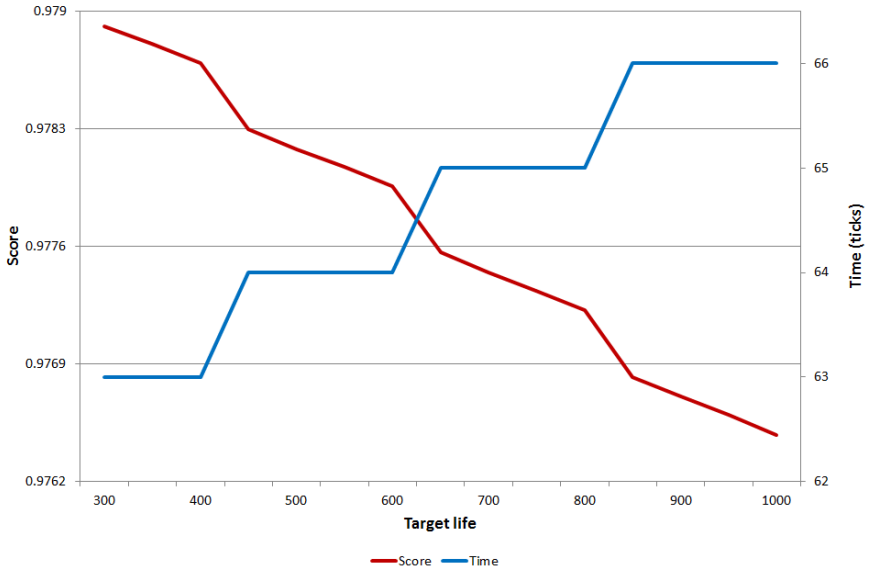


Figure 36: Results for variation of target life l_t.

The resulting graph seems to confirm our expectations, also considering the previous outcome. However, there is a difference: in this test the score and the time lines are not increasing/decreasing like before but following a stepwise trend instead, easily seen especially in the time graph. The explanation to this is that there are certain ranges of values for l_t in which the swarm obtains the same results, dealing to such graph.

In other words in a simple environment such the one studied, there exist "safe ranges" for the target life values, meaning that with the same number of nanorobots we can assume to achieve a good result, even if such values are slightly higher than expected (e.g. with a population of 100 agents and l_t in the interval $[650, 800]$, the final $\alpha - score$ has an average $\mu = 0.9773906$ and a variance $\sigma^2 = 0.000000016118$)[2]

Testing target's life variation we can see that the relation between the UB, LB and score has the same trend already seen: the score follows almost linearly the UB (Figure 37), which decreases because to calculate such bound the value of l_t is subtracted from it. In this graph the LB is not included because its computation is independent from l_t and that makes it constant for all the runs.

Because of the simplicity of the experiment settings, further tests have been considered not so useful.

2 Please remember that $\alpha - score \in [0, 1]$.

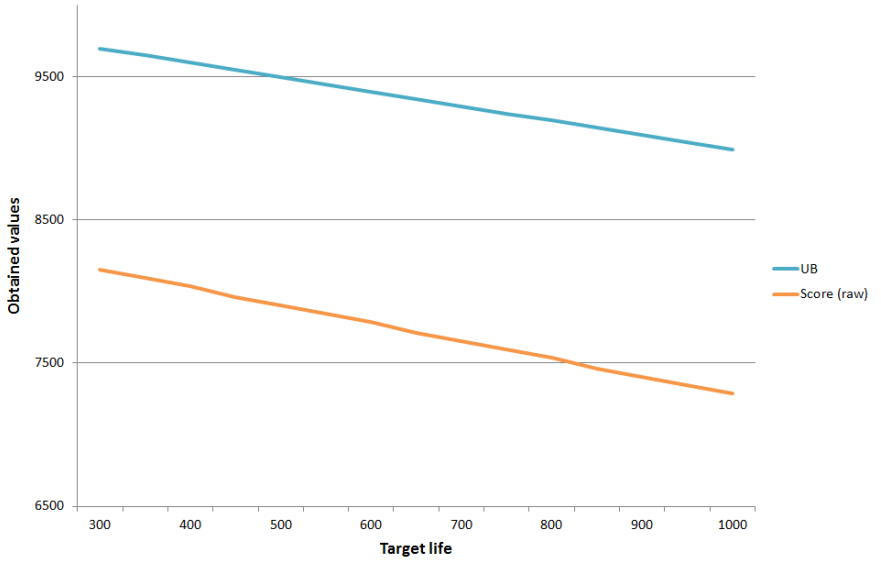


Figure 37: UB and raw score comparison with target life variation.

7.3.2 *Medium*

The second experiment involves a more interesting environment, presenting alternative routes for the nanorobots and, consequentially, the possibility for the target to not be found and eliminated.

However there is still a main loop, to which all the detours connect. The target is placed halfway in such loop, more precisely in one of the two capillary routes.

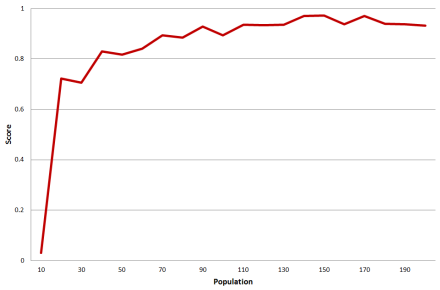
Population study

For the first test we study again the obtained results modifying the population parameter. The number of agents has been increased from 10 to 200. Results are visible in Figure 38.

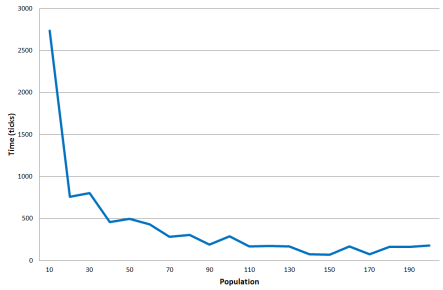
As seen with the *simple* environment, increasing the population deals to better results: the swarm is able to kill the target faster and more efficiently. However this time something new occurred: with a population number equals to 10, which is the least number of agents to consider our group a swarm, the target is not killed before the nanorobots run out of energy. Being this a failure, its $\alpha - score$ present a low value, really close to the LB (see Figure 39).

This situation can be explained by the fact that we have increased the complexity of our environment and, if before 10 agents were still able to find the unavoidable target, now they are not sufficient to explore the entire system and cooperate to achieve their goal.

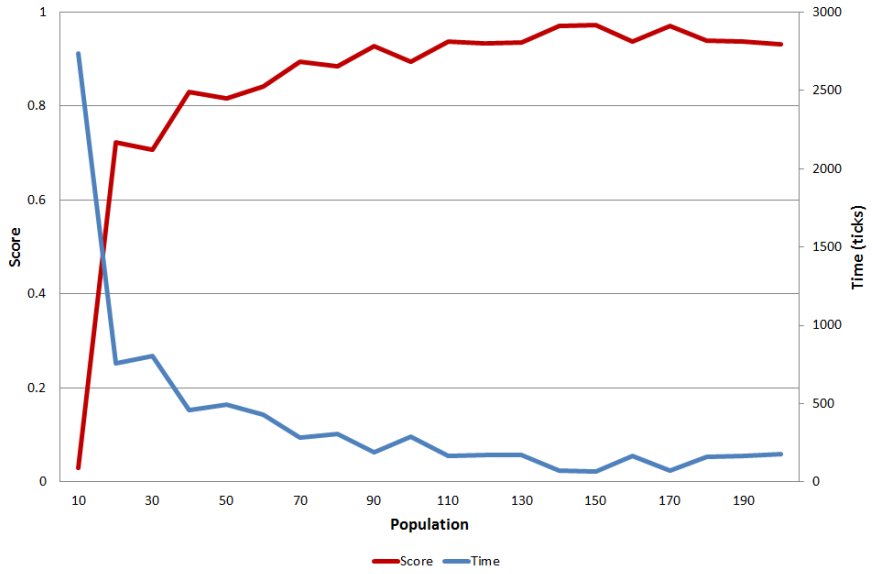
Also, increasing the population over a certain threshold does not improve significantly the obtained score, even if effects of the probability can be seen in the graph.



(a) Score variation.



(b) Time variation.



(c) Combined graphs.

Figure 38: Results for population variation.

Indeed, from this experiment, probability will start to play a big role in the exploration phase: at each branch point each agent can choose which direction to follow, under a certain given probability. This helps the swarm to spread in the vessel system but, at the same time, makes the coordination more complicated.

Communication via CoMa will also start to play an important role in our tests.

If we exclude the failure, the bounds graph still shows the score line following closely the UB (without reaching it). However it does not present an almost linear trend but some fluctuations can be seen.

The environment is still quite simple, but the added complexity starts to affect the obtained results.

As before, UB and LB increase (or decrease) linearly with the population's dimension because of the way they are computed.

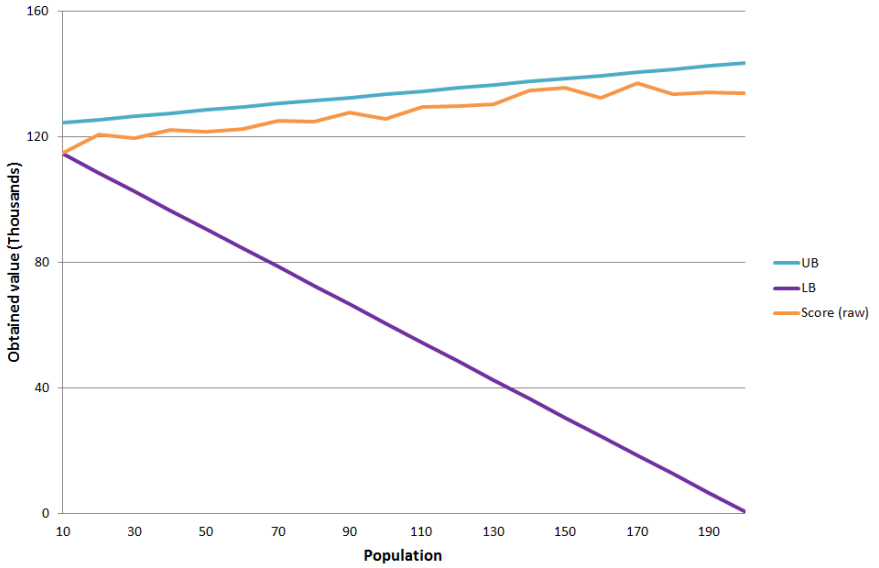


Figure 39: UB, LB and raw score comparison with population variation.

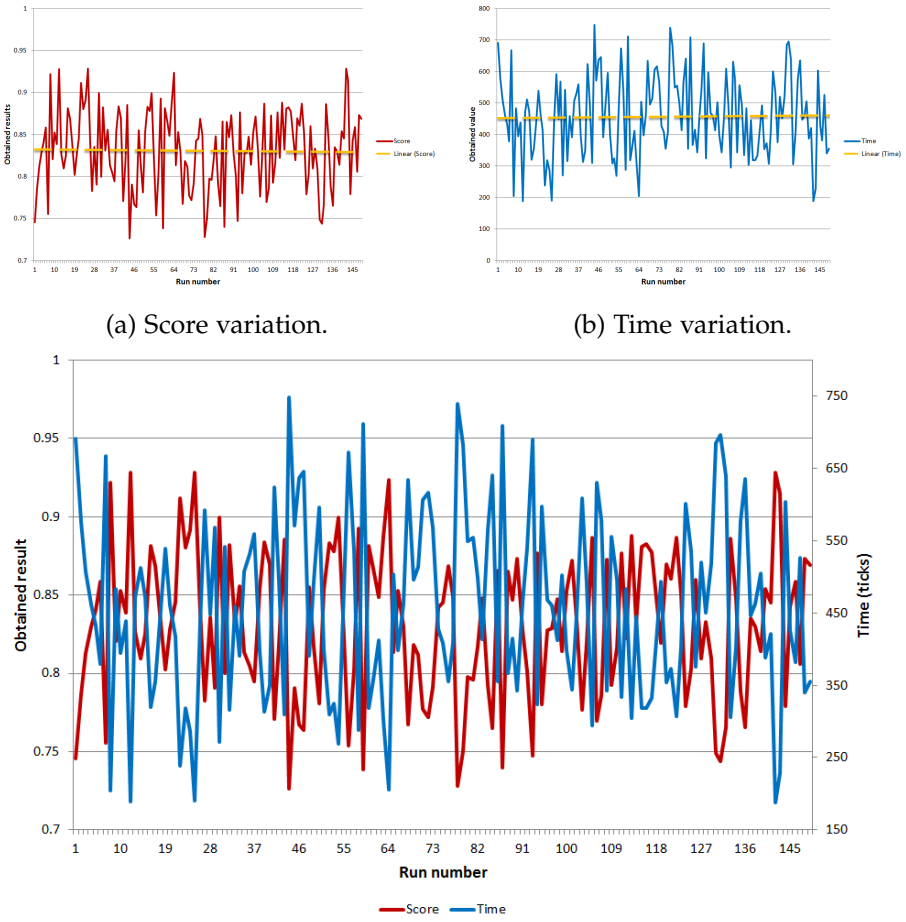
The second test has the purpose to study how probability influences the behaviour of the swarm. So population has been set to the value of 50 (which leads to a $\gamma = \frac{1}{3}$) and the simulation has been run an high number of times to get samples.

The obtained outcome is interesting (Figure 40).
Graph 40a shows the $\alpha - score$ results, which vary a lot but tend to stay in the range $[0.75, 0.9]$. This means that each individual run is very influenced by the chosen paths and probability, but overall the swarm perform very well, with few bad results (this time in 150 runs there was no failure).

Graph 40b shows the time variation. As expected it follows what already seen for the score, also because the two are strictly correlated: the swarm obtaining an high score normally means that the target has been eliminated in a short amount of time. Such correlation is easily seen in Graph 40c.

In both graphs 40a and 40b are visible the trend lines (dashed and in yellow). Such lines are obtained using a linear function and show the trend of the runs: despite of the high variation of values, the it appears to be constant, without showing any increment of decrement of the performances. This confirms the idea that the probability is indeed important but it does not influence the outcome in terms of overall results.

Using a logarithmic or an exponential function gives us the same trend lines.



(a) Score variation.

(b) Time variation.

(c) Combined graphs.

Figure 40: Results for multiple runs with same parameters.

Target study

Next experiment considers again target's life variation. As we expect, as l_t increase, the performances start to worsen: the nanorobots will need a bigger amount of agents and healing drug, leading to an increased consumption of time and a lower score (Figure 41). However such process is not so linear and there are cases in which the swarm achieved a better result despite what we were expecting from the general trend. Again this is a consequence of a better exploration, which is probability-driven.

With this in mind, we can better understand the relation between UB and score shown in Graph 42: the score still tend to decrease with the growth of l_t but big fluctuations are present, sign that probability highly influences every single run. That is why, even with a target with a low life parameter, the swarm can achieve a not very high score, still remaining closer to the UB than to the LB.

In short, now that the environment has some added complexity, the outcome can easily change from run to run, even with the same

Figure 41: Results for variation of target life l_t.

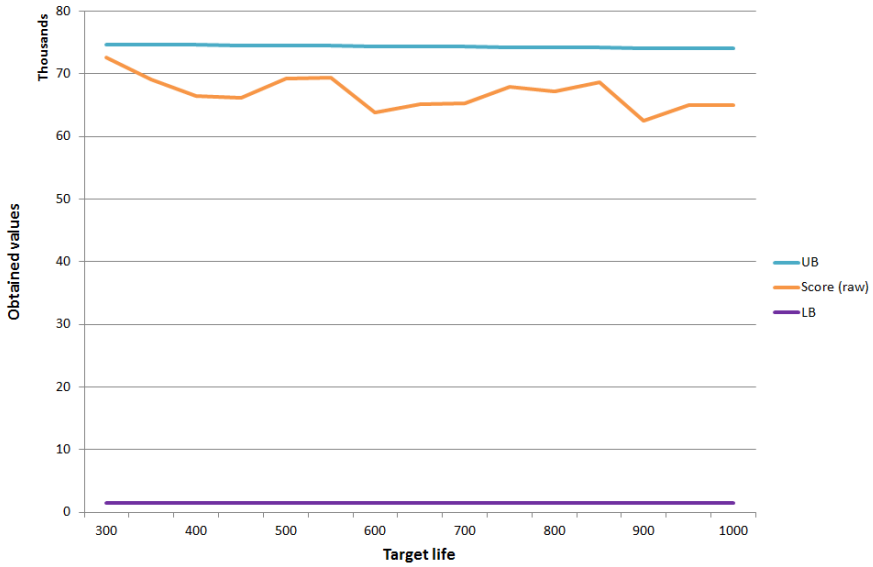


Figure 42: UB and raw score comparison with target's life variation.

parameters setting, but without much difference from the expected result. Please note that the values presented in Graph 42 have been translated by the function $f(x) = x + 65,000$ (i.e. moved by 65,000 units), to avoid negativity. As for *simple* experiment, UB is slowly decreasing and LB is constant (we already discussed why).

CoMa study

The next test involves CoMa instead of target's life. The idea is to study how a different amount of marker influences the behaviour of the swarm.

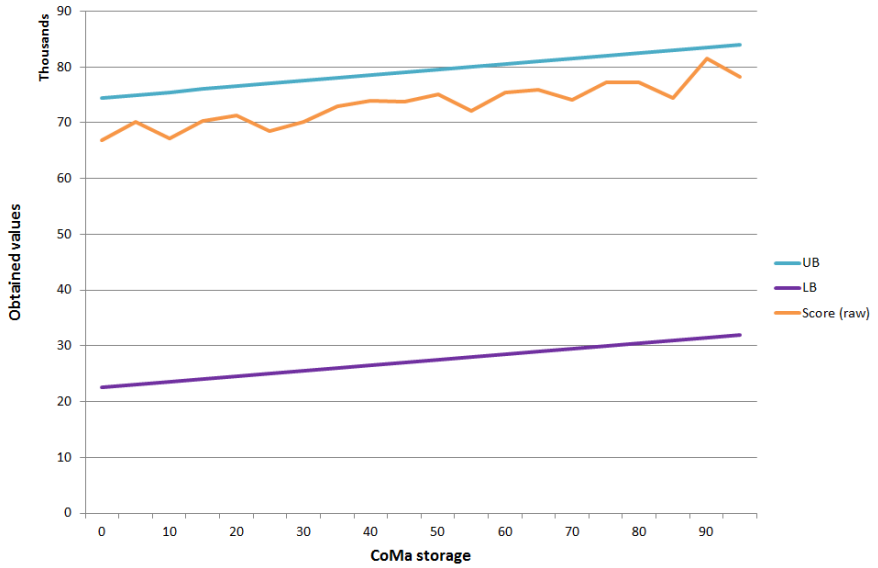


Figure 43: UB and raw score comparison with CoMa variation.

For this experiment the population has been set to the value of 100 for all the runs, same to l_t which is equal to 500 and the marker storage for each nanorobot has been modified in the interval $[0, 100]$, increasing it 5 units at a time.

What we can notice from Graph 43 is that the results follow an increasing trend, with some fluctuations. This means that the swarm performs generally well, obtaining a score that is always close to the UB. Surprisingly, tests involving a very low amount of CoMa register good performances, even if the amount of marker is equal to 0. In such case communication is possible only for a very short distance, or not possible at all, and agents will explore the environment with a random walk technique, without relying on the exchange of information.

The fact that the nanorobots are able to achieve their goal even under such conditions, is most likely a consequence of the limited environment, which do not implement an high number of alternative routes to reach the active site, together with a sufficient energy supply, allowing the agents to explore the vessel system in its entirety.

A second observation is related to the increment that can be seen in the graph: such increment is primarily due to the manual increase of the CoMa parameter and only secondarily to a real improvement in the healing process. All the scoring functions are related to the amount of marker contained in the agents when the target is killed and increasing the overall quantity has as consequence an increment of such term in our evaluations.

For this reason UB and LB are not decreasing, as we were used to, but increasing instead.

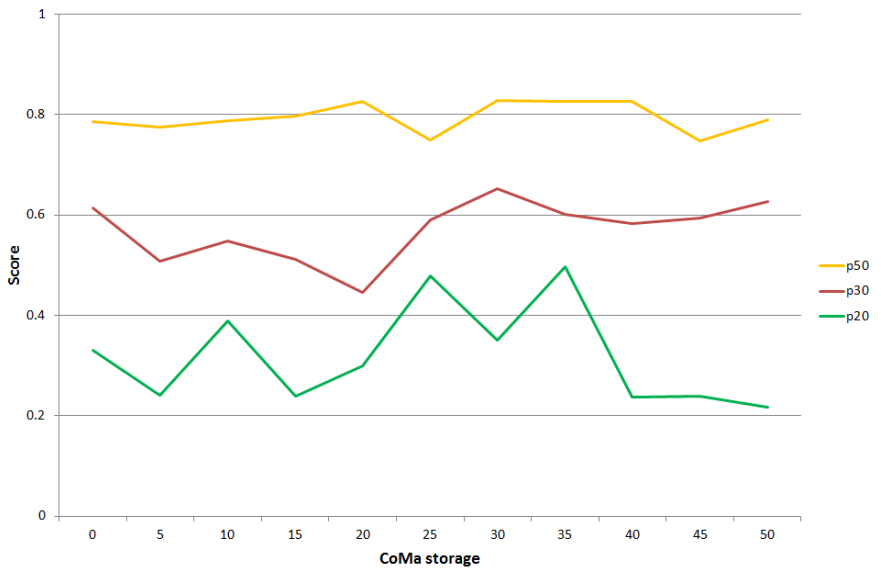


Figure 44: Score variation in response to different amount of CoMa.

To investigate the connection between CoMa and the agents behaviour another test was designed: with fixed number of agents in the environment, the amount of marker stored has been modified in the range $[0, 50]$, as done for the previous test. With the name *p50* we indicate a population of 50 agents and so on. Results are in Graph 44.

Checking the obtained score values, we can see that results for a fixed population seems to not vary with the modification of the CoMa level. There are, of course, variations caused by the probabilistic aspect of the exploration, but the range of such variation is not so high to categorize it as a performance improving trend.

Such results bring to the conclusion that the obtained $\alpha - score$ depends almost only on the population size, making the amount of CoMa not a relevant parameter in the test. In other words, this seems to tell us that communication is not determinant in achieving a good result.

However two aspects have to be taken into account before accept any conclusion: the *medium* environment is indeed more complex than the *simple*, but remains a basic case of study, where the agents are easily able to visit it in its entirety. Second, in this test the amount of drug stored in each nanorobot was not modified and set to the quite high value of 50. This means that each nanorobot can heal $\frac{1}{10}$ of the target full life, making the number of agents needed to eliminate it equal to just 10.

A small number indeed, but close to what would be a real scenario of use for the technology, where the doctor has to inject a swarm carrying an amount of medicine considerably larger than the estimated amount needed to eradicate the cancer cell.

Population & target study

As last experiment for the *medium* environment, two parameters were studied together: population and drug storage.

This experiment is composed by five sub-experiments, each one with a fixed population value for all the runs and variable drug storage. The drug parameter has been varied in the range $[5, 50]$, 5 units at a time. As usual time and score have been registered and are visible in Figure 45.

Population has been subdivided in three test settings:

- **p100** : number of agents in the environment = 100, $\gamma = 0.735$
- **p50** : number of agents in the environment = 50, $\gamma = 0.368$
- **p30** : number of agents in the environment = 30, $\gamma = 0.221$

Let us start considering the score results: as expected the more nanorobots involved, the better the resulting score is. Each line, resulting from runs with a fixed population, shows how the quantity of medical drug influences the run outcome.

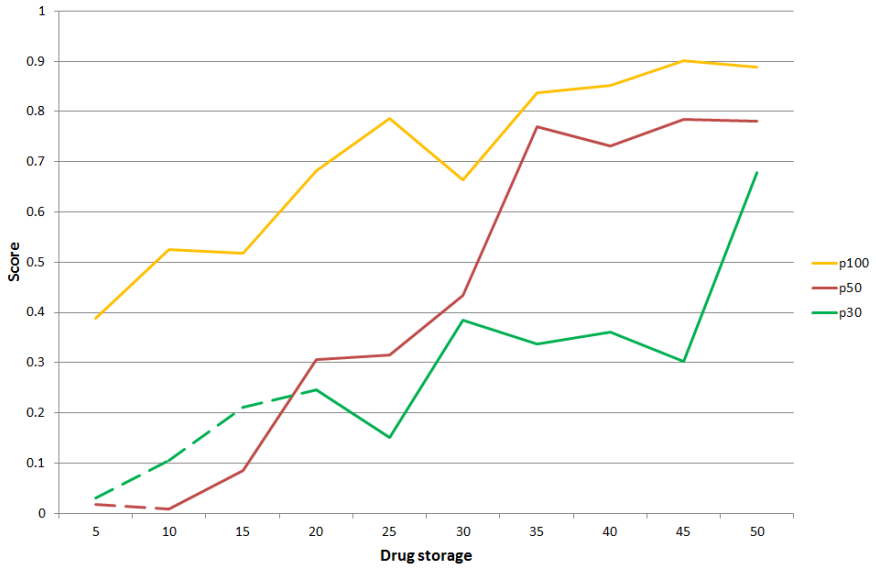
First of all, the lines for *p30* and *p50* show a dotted initial part, which correspond to a low $\alpha - score$: as we expect that displays a failure but what has to be considered is that such failure is a consequence of the fact that with that number of agents carrying the selected amount of drug is impossible to heal the target, just because the total drug in the environment is not enough. Simple mathematics.

We should then consider the results obtained for such population settings beginning from the point where the nanorobots are actually able to fulfil their task.

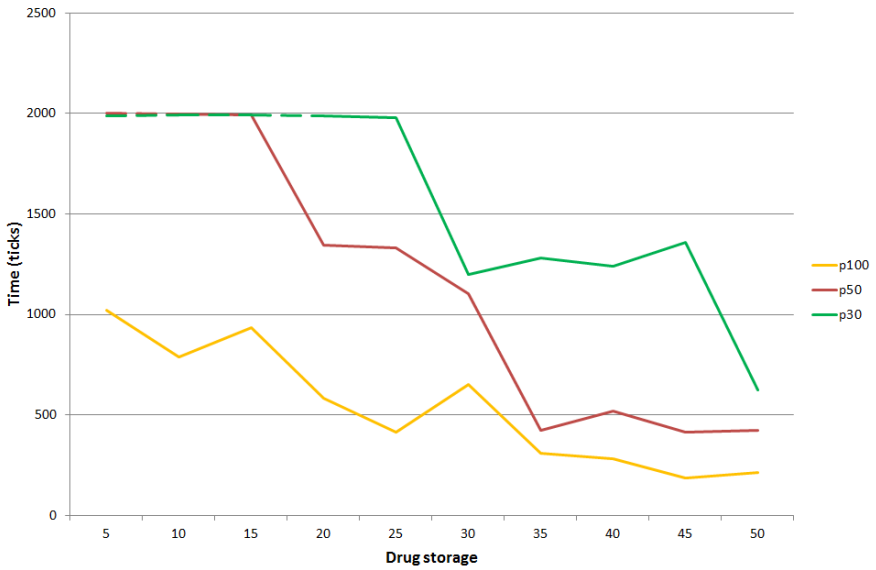
After this introduction we can start to analyse the results. Not surprisingly the score improves as we allow the agents to carry more drug. This means that less agents are needed to kill the target, making the process easier and faster, since less of them have to find it. The improvement showed follows an almost linear trend, increasing a lot at each step.

This confirms the idea that the medicine storage is one of the most important parameters in our study and have to have the priority over the others when implementing the system.

Observations made for the score find confirmation in the graph of the used time (Figure 45b), which better shows how *p30* and *p50* fail



(a) Score variation.



(b) Time variation.

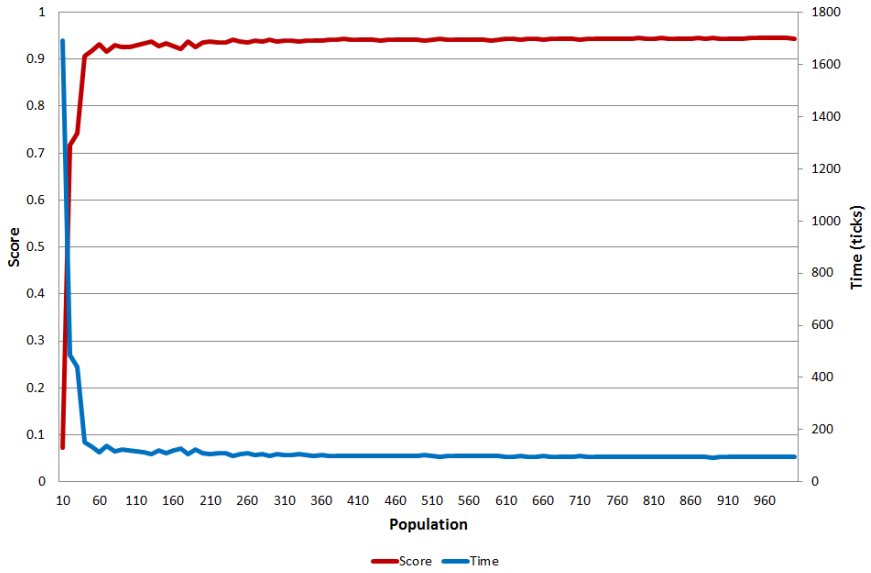
Figure 45: Score&Time variation in response to different amount of drug.

when the carried drug is not sufficient.

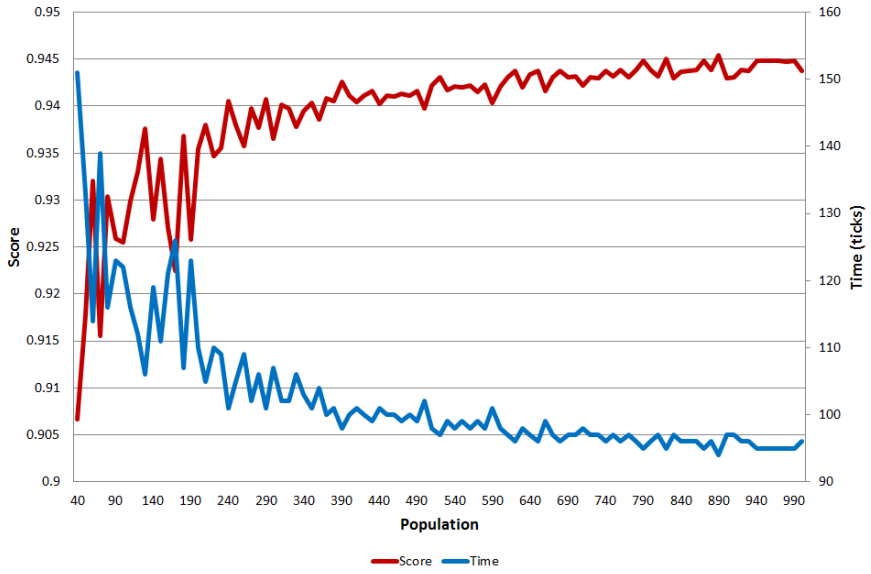
If we consider the score of 0.6 as a threshold to distinguish a bad from a good results, which is plausible for this environment, we can identify the settings performing better. With a population greater than 30 the swarm achieves easily good results and, once again, such population present a γ parameter close to $\frac{1}{3}$. Moreover, the amount of drug stored in each agent has not to be so high to achieve a good score: even with a value around 25, which is half of the value normally used in the *medium* experiment, the swarm is able to complete its task in a reasonable amount of time.

The number of agents is what makes the difference, and this is quite straight forward because having more agents exploring the environment increases the probability to find the target earlier. Anyway, since the number of entities that can be injected in a human body is not infinite, the population of nanorobots will always be limited, making the amount of the medical drug stored another important parameter to consider.

7.3.3 *Complex*



(a) Score&Time variation.



(b) Score&Time variation (detail).

Figure 46: Results for population variation.

For the third experiment we will use a more complex environment: with a total of 536 patches, this is environment contains an high number of alternative paths and connections, creating many sub-loops inside the main loop. Exploration in this environment becomes more interesting and complex, forcing the agents to cooperate even more.

In this experiment the target is placed in a vein vessel, not so far from the capillary network, in one of the alternative paths that branch

from it. From the four "exits" of the capillary system, only two guarantee access to the active spot, without considering the possibility of going back, of course.

Population study

Our first test is once again about population. This time the parameter has been increased from 10 to 1000, adding 10 agents each step. Results are visible in Figure 46.

The graph reports just the resulting time and score combined but it is quite clear that a low number of agents is not enough for the new environment: not surprisingly a swarm composed of only 10 brave nanorobots was not able to eliminate the target. They managed to find it but, since of the low number, the participation of all of them was required to heal the target and the environment is just too vast to be able to coordinate with success. First failure for the swarm (if we can call it like that).

The unsuccessful result was indeed expected. What was a surprise are the results obtained at the next steps of our test process: already with 20 nanorobots the target is found and killed and from 40 agents the $\alpha - score$ already reach the value of 0.9, which is indeed a very good result.
Graph 46b shows results with more details, starting from population 40. This helps to notice the good performance of the swarm but shows also that the such score increment slower significantly around population value of 600.

Increasing the population shows a boost in performance in the beginning, but after a certain value it does not affect the results so much to justify the presence of such elevated number of agents.

To confirm our reasoning, Figure 47 shows that incrementing the population deals to a variation of the UB and LB, with the score following closely the first, as expected. However, it slows down its incremental trend, increasing the gap with the UB as we increase the population parameter. Please note that the values presented in the graph have been translated by the function $f(x) = x + 400,000$ to avoid negativity in the representation.

The second test has the purpose to study how probability influences our results. To do so, population has been fixed to the value of 175 (which leads to $\gamma = \frac{1}{3}$) and the simulation has been run 150 times. Results in Figure 48.

Not surprisingly the graph tell us that to an high score corresponds a low time. Also we can notice that probability plays a primary role

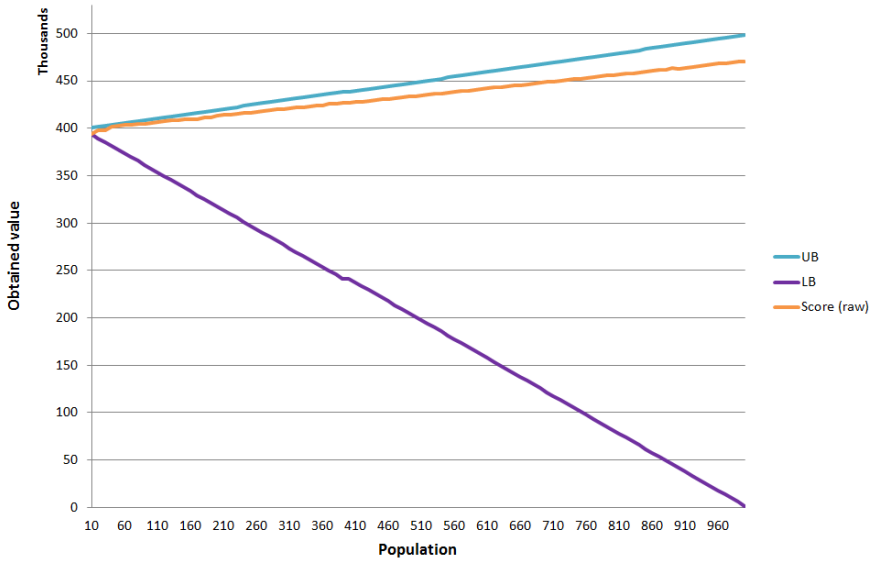


Figure 47: UB, LB and raw score comparison with population variation.

in the quality of a single run: with an unchanged parameter setting, the resulting trend for both score and time is in fact really jagged though maintaining an overall linear trend. Despite the influence of probability affecting the exploration phase, the swarm shows good performances, which tell us that for a generic run we can reasonably expect the outcome to be good.

The $\alpha - score$ resulting is always in the interval $[0.92, 0.94]$ which, considering the number of run, is an amazing result. No failure have been registered in the test. The trend line, in yellow in the Graph 48a, seems to be constant but, after a closer look, a slow diminution can be notice. This can be considered as a mere result of some not very good runs and should not make us thinking that the score result will decrease on the long run. The registered reduction is on the order of 0.001 points, no something that should worry us.

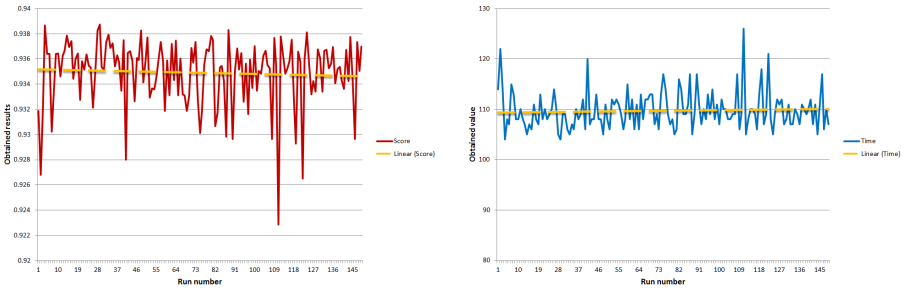
Time, always specular to score, shows an almost imperceptible increasing trend (Graph 48b): consideration done for the score can be applied to the time results as well. This test confirms again that probability influences each single run but not the final outcome.

Target study

The second test studies target's life variation.

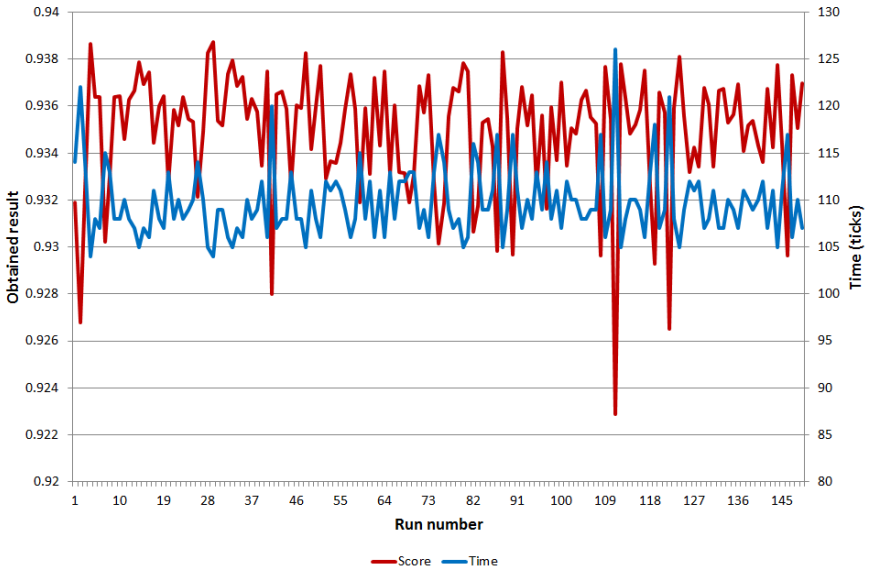
For this test population has been fixed to out usual value of 175, to achieve the $\gamma = \frac{1}{3}$ and l_t has been incremented 50 units per step, in the range $[300, 1000]$.

Differently than other experiments, increasing l_t do not decrease the performance immediately: the worsening trend can be only seen



(a) Score variation.

(b) Time variation.



(c) Combined graphs.

Figure 48: Results for multiple runs with same parameters.

after some increments and at the end of our test, with $l_t = 1000$, the performance has decreased by only 0.01 points.

Such results is a consequence of the good balance of amount of drug stored in each agent and their population.

Comparing the score with its UB confirms what observed (Graph 50).

Both decrease and this is due to the increment of the l_t parameter, which is used to evaluate some terms in the scoring function. In the graph the LB is not shown because it does not depend on the target life, making it constant for all the runs.

Again the result of the randomness in the exploration adds fluctuations in the general trend for this test.

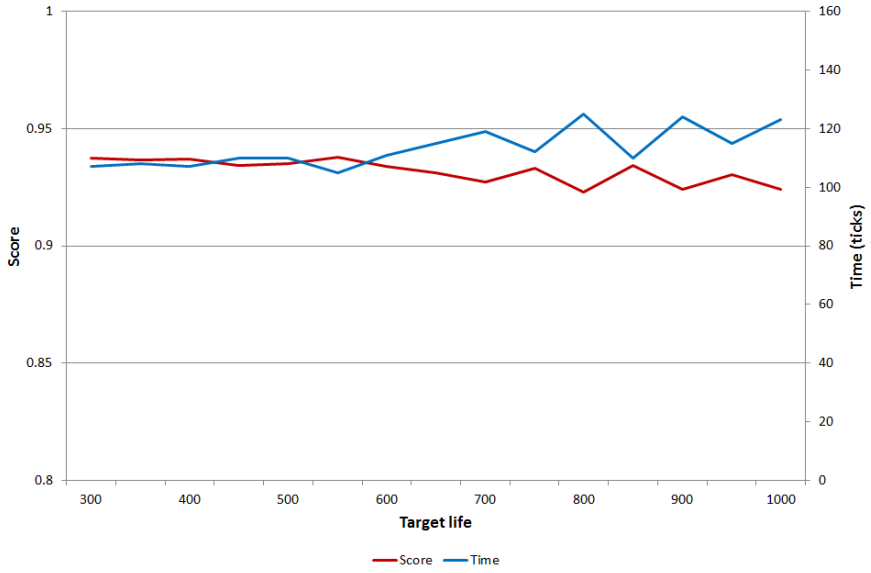


Figure 49: Results for variation of target life l_t.

CoMa study

Third test studies CoMa variations and their effect on the simulation.

What we want to study is the relation between CoMa and the final score in a fairly wide environment such as the *complex*.

The test is composed by five sub-tests, each one with a different fixed value for the population. We will progressively increase the marker storage, starting from 0 until reaching the value of 50 (the one used in most of the experiments so far). Unlike the test using the *medium* environment, the population is now subdivided in relation to its γ value instead of the mere number of agents.

The subdivision results as follows:

- **p175** : number of agents in the environment = 175, $\gamma = \frac{1}{3}$
- **p110** : number of agents in the environment = 110, $\gamma = \frac{1}{5}$
- **p55** : number of agents in the environment = 55, $\gamma = \frac{1}{10}$
- **p30** : number of agents in the environment = 30, $\gamma = \frac{1}{20}$

The results obtained from the test are visible in Graph 51.

The first thing to notice is that all the runs obtain excellent result, scoring almost always more than 0.7 points. Tests *p175*, *p110* and *p55* in particular never score below 0.9, showing no signs of being affected by the lack of CoMa. This proofs once again that having a large marker supply does not makes the difference and, with an high number of agents exploring the environment, we can always expect good results.

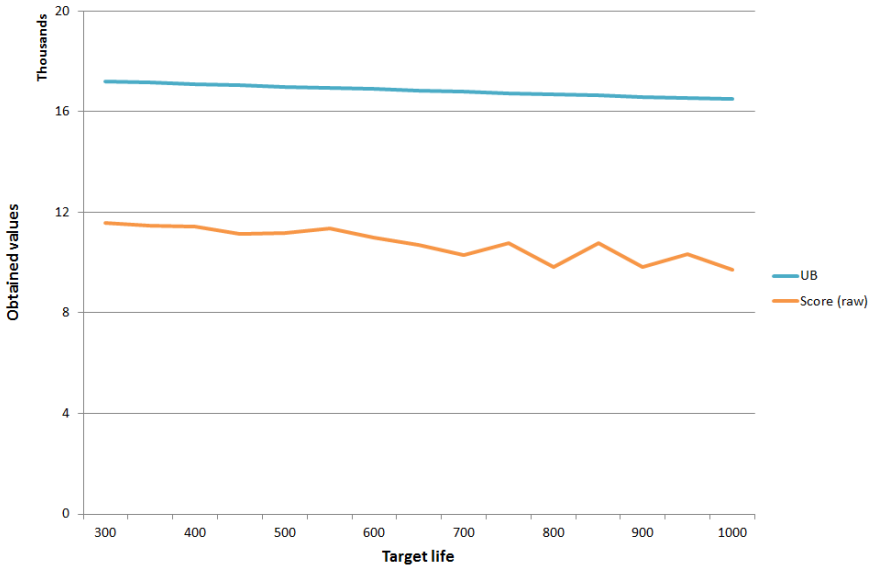


Figure 50: UB and raw score comparison with target's life variation.

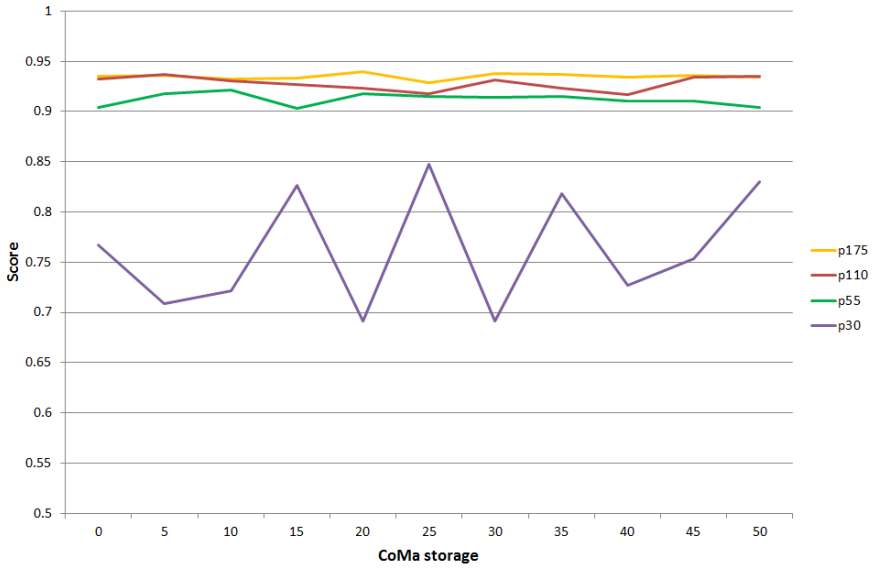


Figure 51: Score variation in response to different amount of CoMa.

The trends however are not of increment, but tend to be stationary for all the runs.

The same results were found in the *medium* experiment, so we can consider this a confirmation of our supposition.

For what concerns the *p30* run, it indeed presents some variations which are correlated more to poor choices in the exploration phase than to a correlation with the marker available. Overall it performs fairly well, as the others do: clearly its scores are lower than the others

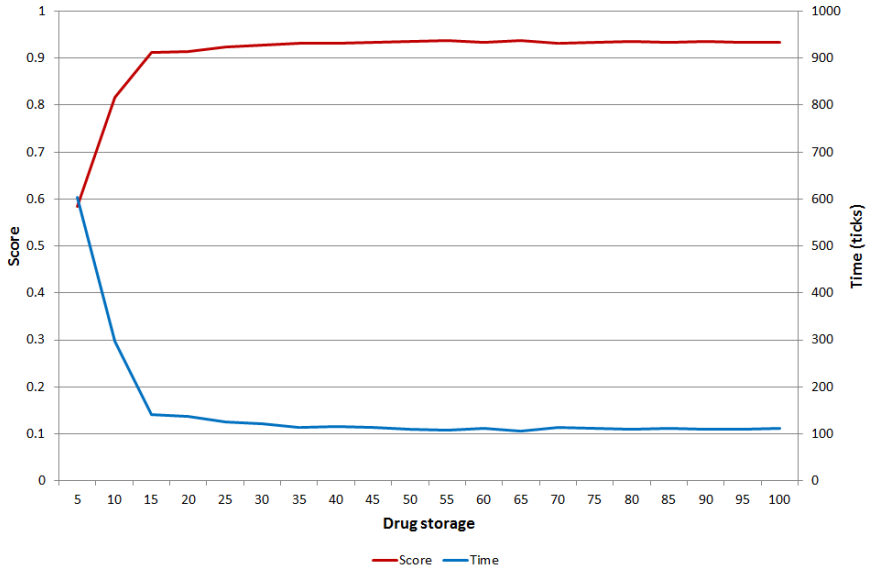


Figure 52: Score variation in response to different capacity of the Drug storage.

because of the difference in population. Once again the test underline how population affect greatly the quality of the outcome.

Population & target study

As last experiment for the *complex* environment the variation of population and drug storage was studied.

Before proceeding with the comparison of these two parameter and their effects in the simulation it is better to understand how the amount of stored medical drug affects the behaviour of the nanorobots. To do so the population value has been fixed to the well known value of $\gamma = \frac{1}{3}$ and the drug storage parameter has been modified in the interval $[5, 100]$, with increments of 5 units each step. The case with drug storage $= 0$ has been excluded from the study.

What stand out from the results (Graph 52) is that providing a low amount of drug to the nanorobots makes their work more complex: since of the lack of medicine, an higher number of agents have to coordinate and being able to reach the active site. However, already with a small improvement ($10 - 15$ units), the score greatly increase and maintain such level of performance for higher values of the parameter.

Such results tell us that drug storage is a parameter that directly affects the healing process, strictly related to the final resulting $\alpha -$ score, but for a sufficient population it can be kept small enough without a loss of performance.

Next test studies the connection between population and drug storage.

Population has been subdivided as for the study of the CoMa variation: excluding the already observed value of 175, we will fix its value to 110, 55 and 30.

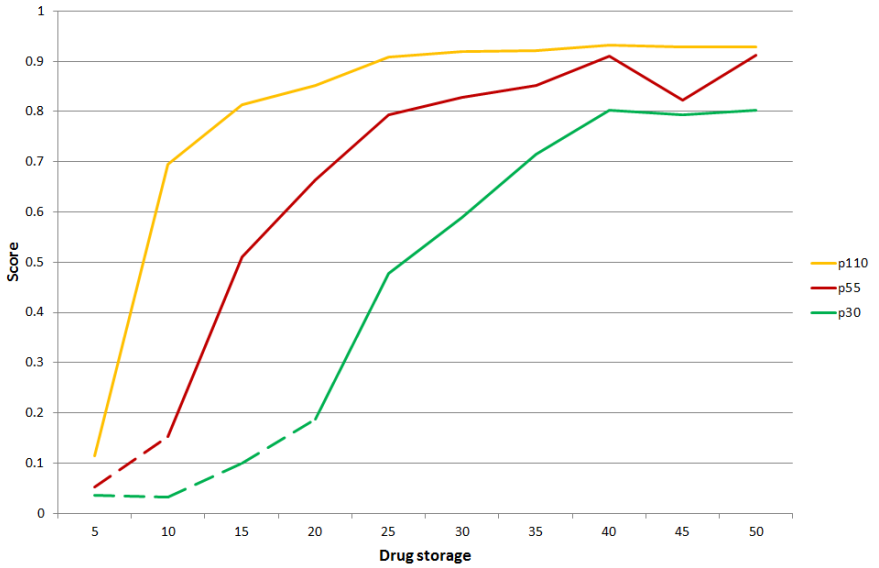


Figure 53: Score variation in response to different amount of drug.

With such population samples we will proceed to change the drug storage level from 5 to 50 and study the outcome, shown is Graph 53.

Before starting the analysis there is an observation to do: the lines for *p30* and *p55* show a dotted initial part, which correspond to a low $\alpha - \text{score}$. That part represents the situation in which the quantity of drug is not sufficient to eradicate the target: since the number of agents is small, assigning each one a quantity too low of medicine does not reach the amount of target life. Another option would have been to assign a fixed warning value to the score in such conditions or to not test it. However we think that the results in a failure can still be interesting and tell us useful information. This explains why that part of the resulting lines has been marked in such way.

From the score graph 53 we can notice that a small amount of drug corresponds in all cases to a poor performance: each agent can heal a small part of the target, leading to a forced presence of an elevated number of agents in the active site. This can happen only if good communication and coordination are achieved.

However this is true only until what we could consider a first threshold level for drug storage: after that point the results improves drastically and keeps increasing until reaching a second threshold, af-

ter which the results do not seem to improve. Or at least not with the previous degree of improvement.

The amount of such increment depends on the population: **p110** needs just one step of 5 storage capacity to raise the score from 0.1 to 0.7, while **p30** needs around 15 units to reach the same performance level.

Sooner or later, all the runs reach the second threshold, where the score start to stabilize and follow an almost linear trend. Trend which seems to be still increasing, but so slowly it looks unchanged. This situation exists for all possible values of the population parameter and has to be identified in order to limit the amount of unnecessary stored drug.

In conclusion, this last experiment confirmed the strong correlation between the two parameters we are studying and showed how good results can be achieved with a good balance of the two.

8

CONCLUSION

From the analysis performed and the results obtained we can get many interesting conclusions.

First observation is about the importance of the environment.

The dimension of the environment makes the difference when we want to study the behaviour of our swam of nanorobots: implementing a small simple system will not tell us anything useful because of the lack of challenge for the agents. Instead, in a fairly complex scenario, we can study how each parameter in the simulation is related to the observed behaviour, and modify them to discover new results.

Such complexity can be decided by the user of the software but remains still bounded to the dimension of the input environment created with the *Builder* software. The choice for that process to have the user defining the direction of each edge in the circulatory model was a consequence of the idea that the software can not determine if a connection follows one direction or its opposite, with the result that the environment could not be the one the user wanted to implement. Having the user defining the environment in all its aspects give him/her total control over the input model for the simulation.

A second observation is about the implemented nanorobots's model. Since this technology is at its beginning, many ideas have been developed but are far from a real implementation. Some of them are really promising, others are just an hypothetical solution for the problems arisen.

All the aspect of the implemented model have been decided thinking about what nanotechnology can really achieve in the next years, always choosing the most plausible solution to implement.

The experiments studied have shown some interesting results.

The most important is the correlation between the number of agents involved in the process and the amount of medical drug stored in them. Since we do not have an infinite number of agents available, because of the limited of external entities that can be injected in a patient body, the performance of the swarm depends on the amount of drug each agent can release at the active site, where the target is located.

Having a low amount of medical drug bring a situation where the an high number of agents have to reach the active site, forcing the swarm to rely only on communication and a good exploration phase.

On the other hand, having too much of this medical drug available results in an high amount of unused substance and has been proved to not increase performance over a certain level. What should be done is find the value after that the results stop to improve and assign that amount of drug to each nanorobot. Experiments have shown that such value can also be small, but has always to be a few times bigger than the target's life.

Population, as the most important parameter with medicine, has also to be chosen after a study of what will be the environment the swarm will have to operate in: an easy solution would be the use of an high number of agents, which makes the search for the target faster, allows each robot to carry less drug and eliminate the target in less time. The problem is that such approach is impracticable in a real application.

Experiments have also proved that a value of population too high do not significantly improve the results.

The general rule should be that the lower the number of nanorobots in the environment, the bigger the drug storage should be, in order to allow the few of them able to spot the target to still be able to release a relevant amount of medicine.

The ideal number of agents and quantity of medicine has to be decided in relation to the extension of the environment and the dimension target cancer cell.

Another important outcome from our experiments concerns the Communication Marker: it has been proved to be important for the coordination and communication between the agent but, at the same time, it is not so decisive to obtain a good result. Experiments have proved that increasing the quantity of CoMa do not correspond to achieve better results: since the agents rely most on the time on a random walk exploration, communication becomes important only when the target is found and the number of agents is not sufficient to eliminate it.

Communication marker can be considered indeed a secondary parameter, correlated to population and drug storage, but not as important in achieving better performance and showed that using chemical signals for communication works good locally, but tends to be not very effective when the swarm is scattered in the environment and we need to gather the higher number of agents in one place in a short time.

This is also a consequence of the fact that marker tends to fade away before being detected by other agents. This can not be avoided because one of our first requisites was to leave as least possible traces in the patient body.

Under these considerations we can conclude that an ideal nanorobot has a quite small CoMa storage, leaving more space for a better energy supply and sensors. Drug storage can also be reduced, but has to be big enough in relation to population and cancer cell expected dimension.

As an extension of this work, constraints can be added to parameters as energy supply, drug storage and population and study the outcome. Target resistance to medical drug can also be taken into account and studied.

In conclusion, this study gave me the possibility to learn and study many interesting aspects of a technology at its early stage, but showing already many promising aspects for the near future and I firmly hope that science will be able to achieve such results and change the actual concept of diseases treatment.

Part III

APPENDIX

A

TOOLS

A.1 NETLOGO

A.1.1 *Presentation*

NetLogo is a programmable modelling environment for simulating natural and social phenomena. It was authored by Uri Wilensky in 1999 and has been in continuous development ever since at the Center for Connected Learning and Computer-Based Modelling. It was designed in the spirit of the Logo programming language to be "low threshold and no ceiling" that is to enable easy entry by novices and yet meet the needs of high powered users.

NetLogo is particularly well suited for modelling complex systems developing over time and the exploration of emergent phenomena. Modellers can give instructions to hundreds or thousands of "agents" all operating independently. This makes it possible to explore the connection between the micro-level behaviour of individuals and the macro-level patterns that emerge from their interaction. It also comes with the Models Library, a large collection of pre-written simulations that can be used and modified. These simulations address content areas in the natural and social sciences including biology and medicine, physics and chemistry, mathematics and computer science, and economics and social psychology.

NetLogo is written in Scala and Java and runs on the Java virtual machine, so it works on all major platforms (Mac, Windows, Linux, et al). It is run as a standalone application. Models can be run as Java applets in a web browser.

Features

- System
  - Free, Open source
  - Cross-platform: runs on Mac, Windows, Linux, et al
- Programming
  - Fully programmable
  - Approachable syntax

  - Language is Logo dialect extended to support agents
  - Mobile agents (turtles) move over a grid of stationary agents (patches)
  - Link agents connect turtles to make networks, graphs, and aggregates
  - Large vocabulary of built-in language primitives
  - Double precision floating point math
  - First-class function values (aka tasks, closures, lambda)
  - Runs are reproducible cross-platform
- Environment
  - Command center for on-the-fly interaction
  - Interface builder w/ buttons, sliders, switches, choosers, monitors, text boxes, notes, output area
  - Agent monitors for inspecting and controlling agents
  - Export and import functions (export data, save and restore state of model)
  - NetLogo 3D for modelling 3D worlds
- Display and visualization
  - Line, bar, and scatter plots
  - Speed slider lets you fast forward your model or see it in slow motion
  - View your model in either 2D and 3D
  - Scalable and rotatable vector shapes
  - Turtle and patch labels
- Web
  - Models can be saved as applets to be embedded in web pages
- APIs
  - controlling API allows embedding NetLogo in a script or application
  - extensions API allows adding new commands and reporters to the NetLogo language

A.1.2 *Programming with NetLogo*

This section gives a brief description of the NetLogo programming language trying to focus on each part that forms it. We will examine first the main elements at the base language and then a brief guide about how to use it will be given.

AGENTS

The NetLogo world is made up of agents. Agents are beings that can follow instructions. There are four types of agents: *turtles*, *patches*, *links*, and the *observer*.

Turtles are agents that move around in the world. The world is two dimensional and is divided up into a grid of patches. Each patch is a square piece of "ground" over which turtles can move. Links are agents that connect two turtles. The observer does not have a location, it can be imagined as an entity that looks over the world of turtles and patches. However it does not observe passively: it gives instructions to the other agents.

The agent that can generate new turtles are the observer and the patches. Despite the fact that they can not move, they can be considered as "alive" as turtles. Patches have coordinates. The patch at coordinates $(0, 0)$ is called the origin and the coordinates of the other patches are the horizontal and vertical distances from this one. We call the patch's coordinates `pxcor` and `pycor`. Just like in the standard mathematical coordinate plane, `pxcor` increases as you move to the right, following the x-axis, and `pycor` increases as you move up. The total number of patches can be determined by the user.

Turtles have coordinates too: `xcor` and `ycor`. A patch's coordinates are always integers, but a turtle's coordinates can have decimals. This means that a turtle can be positioned at any point within its patch. Links do not have coordinates: they has two ends, and each end is a turtle. If either turtle dies, the link dies too. It is represented visually as a line connecting the two turtles.

NetLogo allows you to define different "breeds" of turtles and breeds of links. A breed can be seen as an extension of an agent, a special type of turtle or link. Once a breed has been defined, it can easily behave differently from other breeds or agents.

PROCEDURES

In NetLogo , commands and reporters tell agents what to do. A command is an action for an agent to carry out, resulting in some effect. A reporter is instructions for computing a value, which the agent then "reports" to whoever asked it. Commands and reporter built into NetLogo are called *primitives* and the user-defined are called *procedures*. Each procedure has a name, preceded by the keyword `to` or `to-report`, depending on whether it is a command procedure or a reporter procedure. The keyword `end` marks the end of the commands in the procedure.

Once a procedure is defined, it can be used elsewhere in the program.

Variables

We can distinguish between two main types of variables: *agent* variables and *local* variables.

Agent variables are used to store a value in an agent. They can be a global variable, a turtle variable, a patch variable, or a link variable. If a variable is a global variable, there is only one value for the variable, and every agent can access it: it can be seen as a variable belonging to the observer. Turtle, patch, and link variables are different: each turtle has its own value for every turtle variable and the same goes for patches and links.

Some variables are built into NetLogo . For example, all turtles and links have a `color` variable, which simply represent the color of the turtle. Other built-in variables are the coordinate variables and more.

A local variable is defined and used only in the context of a particular procedure or part of a procedure: such variable will exist only throughout the procedure and will not be available after its end.

Ticks

In most models, time passes in discrete steps, called "ticks". A built-in counter is included to keep track of how many ticks have passed. Normally the current value of the ticks is shown in the main view. Also the view's updates are related to ticks.

Agentsets

An agentset is exactly what its name implies, a set of agents. An agentset can contain either turtles, patches or links, but not more than one type at once. An agentset is not in any particular order. In fact, it's always in a random order and every time used, the agentset will be in a different random order. This helps to keep the model from treating any particular turtles, patches or links differently from any others (unless it is programmed to do so). Since the order is random every time, no one agent always gets to go first.

The interesting aspect of the agentset concept is that the user can construct agentsets that contain only some turtles, some patches or some links. For example, all the red turtles, or the turtles in the first quadrant that are on a green patch or the links connected to turtle `0`.

NetLogo uses the ask command to give commands to turtles, patches, and links. All code to be run by turtles must be located in a turtle context. The same goes for patches, links, and the observer, except that the user can not ask the observer. Any code that is not inside any ask is by default observer code.

An example of the use of ask in a procedure is given in Listing 2.

Listing 2: NetLogo: "ask" example.

```
to setup
  clear-all
  create-turtles 100            ;; create 100 turtles
  ask turtles
    [ set color red             ;; turn them red
      fd 50 ]                   ;; spread them around
  ask patches
    [ if pxcor > 0              ;; patches on the right side
        [ set pcolor green ] ]  ;; of the view turn green
  reset-ticks
end
```

Usually, the observer uses ask to ask all turtles, all patches or all links to run commands. Since every turtle created has a who number (the first turtle created is number 0, the second turtle number 1, and so forth), ask can also be used to have an individual turtle, patch or link run commands (Listing 3).

Listing 3: NetLogo: Select a turtle.

```
to setup
  clear-all
  crt 3                         ;; make 3 turtles
  ask turtle 0                  ;; tell the first one...
    [ fd 1 ]                    ;; ...to go forward
  ask turtle 1                  ;; tell the second one...
    [ set color green ]         ;; ...to become green
  ask turtle 2                  ;; tell the third one...
    [ rt 90 ]                   ;; ...to turn right
end
```

The user can also select a subset of turtles, or a subset of patches, or a subset of links and ask them to do something. This involves using agentsets. When the user asks a set of agents to run more than one command, each agent must finish before the next agent starts. One agent runs all of the commands, then the next agent runs all of them, and so on.

Besides agentsets, NetLogo implements a *List* structure to store informations. Lists allow for the convenient packaging of information: if some agents carry out a repetitive calculation on multiple variables, it might be easier to have a list variable, instead of multiple number

variables. Several primitives simplify the process of performing the same computation on each value in a list.

Lists can be constant, which values are determined at its creation, or built "on the fly", making use of the values calculated by reporters. Indeed the *list* reporter accepts two other reporters, runs them, and reports the results as a list. An example is given in Listing 3: the given code creates a new list, named random-list, with two random values each time it runs.

Listing 4: NetLogo: Build a list.

```
set random-list list (random 10) (random 20)
```

All the usual list operation, such as changing items or iterate over them, are available on NetLogo 's lists.

NetLogo give also the possibility of changing agents aspect, plotting variables or functions, I/O of files and more. The documentation is available at http://ccl.northwestern.edu/netlogo/docs/.

A.2 ENVIRONMENT FILE FORMAT

The file used by both *Builder* and *Simulation* software has the purpose to represent an hypothetical circulatory system composed by arteries, capillaries and veins. As discussed in section A.1, the NetLogo world is a two dimensional grid of *patches*, like a chessboard. Every patch has its own coordinates in the grid.

Using this system, the environment can be represented simply assigning to each patch a "role":

- ARTERY
  A patch classified as artery will have Type 1 and color *red*;
- CAPILLARY
  A capillary patch has Type 2 and color *purple*;
- VEIN
  Patches classified as veins have Type 3 and color *blue*;
- START POINT
  Only a patch in the grid can be classified as *Start Point*. This patch will have Type 0 and color *lime green*. The swarm of nanorobots will start its exploration from this point;
- WALL
  This patch represent the inside of the patient's body, where the nanorobots can not go. Each patch that does not belong to one of the previous type will be marked as *Wall*. A wall patch will have Type -1 and color *grey*.

In addition to this each patch includes some simulation-related variables:

- Pressure
  Variable that shows the amount of pressure a nanorobot will measure in that patch;
- Flux list
  A list that represent the direction the flux of blood is following. This list has four values, one for each possible direction (north, east, south and west). Each direction variable can assure three values:
  1. WALL if there is a wall patch at that direction;
  2. IN if the blood arrive in the patch coming from that way;
  3. OUT if the blood exits the patch going toward that direction.

  an example of this list can be like ["in" "wall" "out" "wall"] meaning that the blood flux enters the patch from north (the adjacent north patch) and continues to south;

- Drug trace
  This variable shows the amount of drug marker present in the patch. At the beginning of the simulation it values is always 0 for all patches;

- Connections
  The number of adjacent patches is stored in this variable. This number is used by the nanorobot to help the navigation.

In light of what that we have shown up to this point, a patch in the environment file is represented as follow:

$$x\ y\ pc\ t\ p\ fl\ dt\ c \tag{16}$$

where x and y are the patch's coordinate in the world grid, pc is the code representing the color of the patch, t is the type of the patch, p the pressure, and fl, dt and c respectively the Flux list, the Drug trace and the Connections variables.

The environment file also contains description of every edge. An edge is the connection of two nodes, where a node is a patch with 2 or more connections. Defining the blood flux, the system automatically creates and defines a certain number of edges. Each edge is represented as a list containing the coordinates of the involved patches: $[x_1\ y_1\ x_2\ y_2]$.

What can be found inside and environment file is a list of all the edges followed by a list of all the patches (see Listing 5).

Listing 5: Environment file structure.

```
"edges"
 [-3 -17 8 -18]
 [-3 -14 2 -14]
 [4 -8 7 -8]
 [2 -14 4 -11]
 ...
"patches"
 9 20 5 -1 0 0 0 0
 23 -5 5 -1 0 0 0 0
 -12 -1 15 1 83 ["wall" "in" "wall" "out"] 0 2
 1 -19 125 2 28 ["wall" "wall" "out" "in"] 0 2
 ...
```

BIBLIOGRAPHY

[1] B Alberts, A Johnson, J Lewis, and et al. Multipotent stem cells: Blood cell formation. URL https://www.ncbi.nlm.nih.gov/books/NBK26919/table/A4143/.

[2] Howard C. Berg. *Random Walks in Biology*. Princeton University Press, 41 William Street, Princeton, NJ, USA, revised edition, 1993.

[3] Genevogue Biotechnology. Application of nano robots in medicine. URL http://www.bioteck.in/2011/12/application-of-nano-robots-in-medicine.html.

[4] Christian Blum and Daniel Merkle. *Swarm Intelligence: Introduction and Applications*. Springer, 1st edition, 2008.

[5] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm intelligence: from natural to artificial systems*. Oxford University Press, 1999.

[6] A. Cavalcanti. Assembly automation with evolutionary nanorobots and sensor-based control applied to nanomedicine. *IEEE Transactions on Nanotechnology*, 2:82–87, June 2003.

[7] A. Cavalcanti, L. Rosen, R.C. Kretly, M. Rosenfeld, and S. Einav. Nanorobotic challenges in biomedical applications, design and control. *11th IEEE International Conference on Electronics, Circuits and Systems*, pages 447–450, December 2004.

[8] Adriano Cavalcanti, Bijan Shirinzadeh, Mingjun Zhang, and Luiz C. Kretly. Nanorobot hardware architecture for medical defense. *Sensors*, 8:2932–2958, May 2008.

[9] David Chandler. Team develops energy-efficient microchip. *MIT Tech Talk*, 52:1, February 2008.

[10] S. Chandrasekaran and D.F. Hougen. Swarm intelligence for cooperation of bio-nano robots using quorum sensing. *Bio Micro and Nanosystems Conference*, pages 104–104, January 2006.

[11] B. C. Crandall and J. Lewis. *Nanotechnology: Research and Prospectives*. MIT Press, Cambridge, MA, 1992.

[12] diabetesdaily. Blood vessel. URL http://www.diabetesdaily.com/wiki/Blood_vessel.

[13] Francis J. DiSalvo. Thermoelectric cooling and power generation. *Science*, 285:703–706, July 1999.

[14] Marco Dorigo, Thomas Stutzle, and Thomas Sttzle. *Ant Colony Optimization*. Mit Press, 2004.

[15] Richard Fleischer. Fantastic voyage, 1966. URL http://www.imdb.com/title/tt0060397/.

[16] Food and Drug Administration. Us food and drug administration homepage. URL http://www.fda.gov/.

[17] Robert A. Freitas Jr. *Nanomedicine*. Landes Bioscience, 1st edition, 2003.

[18] Robert A. Freitas Jr. Pharmacytes: An ideal vehicle for targeted drug delivery. *Journal of Nanoscience and Nanotechnology*, 6:2769–2775, 2006.

[19] Robert A. Freitas Jr. The ideal gene delivery vector: Chromallocytes, cell repair nanorobots for chromosome replacement therapy. *Journal of Evolution and Technology*, 16:1–97, June 2007.

[20] Simon Garnier, Jacques Gautrais, and Guy Theraulaz. The biological principles of swarm intelligence. *Swarm Intell*, 1:3–31, July 2007.

[21] Jonathan Gilbert. The use of nanorobotics to treat cystic fibrosis, 2010. URL http://www.medlink-uk.org/Site/documents/Nano2011/GilbertJ.pdf.

[22] R. Hariharan and J. Manohar. Nanorobotics as medicament (perfect solution for cancer). *International Conference on Emerging Trends in Robotics and Communication Technologies (INTERACT)*, pages 4–7, December 2010.

[23] Cyber Journals. Cyber journals: Multidisciplinary journals in science and technology. URL http://www.cyberjournals.com.

[24] Dervis Karaboga. An idea based on honey bee swarm for numerical optimization. *Technical Report-TR06*, 2005.

[25] James Kennedy and Russel Eberhart. Particle swarm optimization. *IEEE International Conference on Neural Networks*, 4:1942–1948, 1995.

[26] Richard E. Klabunde. *Cardiovascular Physiology Concepts*. Lippincott Williams & Wilkins, 2nd edition, 2011.

[27] Thiemo Krink. Presentation on swarm intelligence, 2001. URL http://staff.washington.edu/paymana/swarm/krink_01.pdf.

[28] T. Kubik, K. Bogunia-Kubik, and M. Sugisaka. Nanotechnology on duty in medical applications. *Current Pharmaceutical Biotechnology*, 6:17–33, 2005.

[29] Bib Latané. The psychology of social impact. *American Psychologist*, 36:343–356, April 1981.

[30] S.M. Masudur Rahman Al-Arif, N. Quader, A. Mamun Shaon, and K.K. Islam. Sensor based autonomous medical nanorobots: a cure to demyelination. *Journal of Selected Areas in Nanotechnology (JSAN)*, September 2011. URL http://www.cyberjournals.com/Sep2011.html.

[31] Mendeley. Mendeley: a free reference manager and pdf organizer. URL http://www.mendeley.com/.

[32] Efrén Mezura-Montes and Omar Cetina-Domínguez. Empirical analysis of a modified artificial bee colony for constrained numerical optimization. *Applied Mathematics and Computation*, 218: 10943–10973, July 2012.

[33] V. Mithra and J. Bhuvaneshwari. Nanorobots in cancer treatment. *International Conference on Emerging Trends in Robotics and Communication Technologies (INTERACT)*, pages 258–264, December 2010.

[34] T. Nantapat, B. Kaewkamnerdpong, T. Achalakul, and B. Sirinaovakul. Best-so-far abc based nanorobot swarm. *International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, pages 226–229, August 2011.

[35] H. Noji, R. Yasuda, M. Yoshida, and K. Kinosita Jr. Direct observation of the rotation of f1-atpase. *Nature*, 386:299–302, March 1997.

[36] IEEE (Institute of Electrical and Electronics Engineers). Ieeexplore®digital library. URL http://ieeexplore.ieee.org.

[37] The Encyclopedia of Science. Blood vessel. URL http://www.daviddarling.info/encyclopedia/B/blood_vessel.html.

[38] Gorka Orive, Susan K. Tam, José Luis Pedraz, and Jean-Pierre Hallé. Biocompatibility of alginate-poly-l-lysine microcapsules for cell therapy. *Biomaterials*, 27:3691–3700, February 2006.

[39] Greeta A. Patel, Gayatri C. Patel, Ritesh B. Patel, Jayvadan K. Patel, and Madhabhai Patel. Nanorobot: A versatile tool in nanomedicine. *Journal of Drug Targeting*, 14:63–67, February 2006.

[40] Mauricio Perretto and Heitor Silvério Lopes. Reconstruction of phylogenetic trees using the ant colony optimization paradigm. *Genetics and Molecular Research*, 4:581–589, 2005.

[41] Lysle H. Peterson. The dynamics of pulsatile blood flow. *Circulation Research, journal of the American Heart Association*, 2:127–139, 1954.

[42] PubMed. Us national library of medicine. URL http://www.ncbi.nlm.nih.gov/pubmed.

[43] Mark A. Ratner and Daniel Ratner. *Nanotechnology: A Gentle Introduction to the Next Big Idea*. Prentice Hall, Inc., Upper Saddle River, NJ, USA, 1st edition, 2002.

[44] Aristides A. G. Requicha. Nanorobots, nems and nanoassembly. *Proceedings of the IEEE*, pages 1922–1933, November 2003.

[45] L. Ricciardi, I. Pitz, S. F. Al-Sarawi, V. Varadan, and Derek Abbott. Investigation into the future of rfid in biomedical applications. *SPIE–the International Society for Optical Engineering*, 5119: 199–209, 2003.

[46] Stuart J. Russel and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, Inc., Upper Saddle River, NJ, USA, 3rd edition, 2009.

[47] Erol Sahin. Swarm robotics: From sources of inspiration to domains of application. *Swarm Robotics Workshop: State-of-the-art Survey*, 3342:10–20, 2005.

[48] TD. Seeley. When is self-organization used in biological systems? *Biological Bulletin*, 202:314–318, 2002.

[49] CancerHelp UK. How a cancer spreads. URL http://cancerhelp.cancerresearchuk.org/about-cancer/what-is-cancer/grow/how-a-cancer-spreads.

[50] B. Watson, J. Friend, and L. Yeo. Piezoelectric ultrasonic resonant motor with stator diameter less than 250 µm: the proteus motor. *Journal of micromechanics and microengineering*, January 2009. URL stacks.iop.org/JMM/19/022001.

[51] Xin-She Yang. Firefly algorithms for multimodal optimization. *Stochastic Algorithms: Foundations and Applications*, 5792:169–178, 2009.

[52] Michael Makoto Zimmer. Model characteristics and properties of nanorobots in the bloodstream, April 2005.