

# DEM Centre Users' Guide

Istvan Elek

2024

# Contents

|          |                                  |          |
|----------|----------------------------------|----------|
| <b>1</b> | <b>Preface</b>                   | <b>3</b> |
| <b>2</b> | <b>Installation</b>              | <b>3</b> |
| <b>3</b> | <b>DC</b>                        | <b>3</b> |
| <b>4</b> | <b>DCDemo</b>                    | <b>4</b> |
| <b>5</b> | <b>DCMaster</b>                  | <b>4</b> |
| 5.1      | Built-in analyser . . . . .      | 9        |
| <b>6</b> | <b>DCAnalyser</b>                | <b>9</b> |
| 6.1      | Load Db file menu . . . . .      | 10       |
| 6.2      | Sql menu . . . . .               | 10       |
| 6.3      | Diagram menu . . . . .           | 14       |
| 6.4      | Discovered fields menu . . . . . | 14       |
| 6.5      | Generations menu . . . . .       | 16       |

# 1 Preface

DEM centre is a program package written in C#, which is the heart of digital evolution. You can create any size of labyrinths with any DEM workers and wumpuses, traps and gold. The system helps you to understand what happened to evolutionary workers in this peculiar world. The system saves all events and workers into a Postgres database, so PostgreSQL 9.6 or later version has to be installed previously. PgAdmin is optional, but very useful tool, so it is also suggested to install. Use the Master to start thousands of DEM workers to discover the world. Start Analyser in order to understand the events.

# 2 Installation

1. Download dc.zip
2. copy dc.zip where you want to run from
3. unzip it
4. start dc.exe

# 3 DC

DC.exe is a frame program, which organizes the components of DEM Centre. You can make the following (Figure 1):

- You can start DCDemo.exe, which demonstrates graphically the movements of DEM entities (workers) in a small labyrinth. There is neither database connection nor knowledge base creation in this case.
- If you are going to make simulations, start DCMaster.exe.
- If you already have simulation results, you can use DCAnalyser.exe to analyse databases, to understand what has happened to evolutionary entities, and what the fate of DEM workers is.

The easiest way to start click to DC.exe, and start further modules from here. Optionally you can start DCMaster.exe if you want to use the simulator or start DCAnalyser.exe if you want to analyse simulation results.

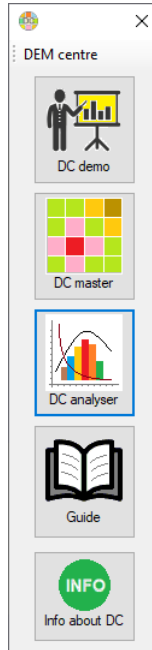


Figure 1: DC Centre form: Demo, Master, Analyser modules can be launched from here.

## 4 DCDemo

DCDemo.exe demonstrates graphically the movements of DEM entities (workers) in a small labyrinth (Figure 2). There is no database connection and knowledge base creation in this case. Here you can create workers, any size of labyrinth with many energy sinks and energy sources. Do not create too much (suggest only one), because the picture may become confusing. If a worker has died the live flag is emphasized with black colour. If the worker found an energy source, the live flag became green, the 'gotcha' flag turns *true* emphasized with green colour.

## 5 DCMaster

If you are going to make simulations, start DCMaster.exe (Figure 3). Here you can create a new labyrinth or import existing ones, create workers, and set or look at parameters. You can set the simulation parameters, such as learning workers, merging knowledge when workers are in the same fields at the same time, start position or random start position, setting offspring's start position to birthplace, random death, saving simulation results to an



Figure 2: DCDemo: A labyrinth with legend and movements can be seen. Numbers symbolize the step number when a certain field is reached.

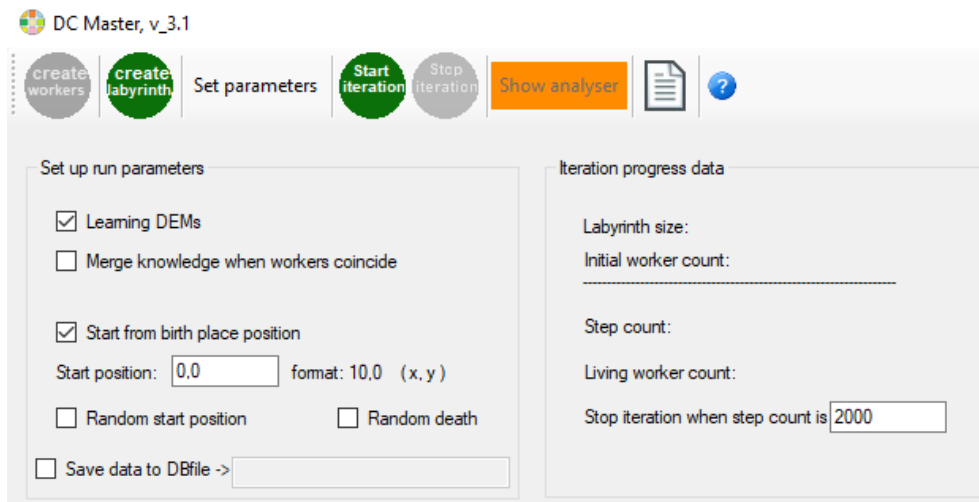


Figure 3: If you are going to make simulations, start DCMaster.exe (Figure 3). Here you can create a new labyrinth or import existing ones, create workers, and set or look at parameters. You can set the simulation parameters, such as learning workers, merging knowledge when workers are in the same fields at the same time, start position or random start position, setting offspring's start position to birthplace, random death, saving simulation results to an SQLite database file

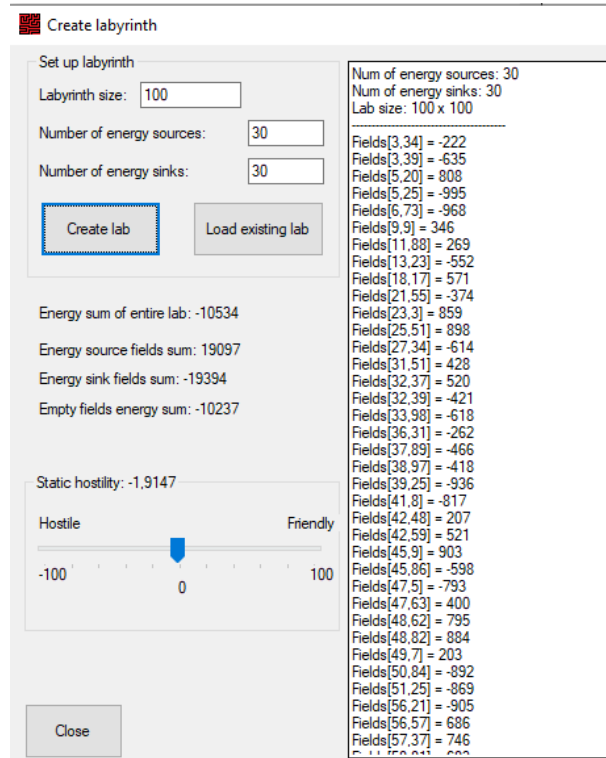


Figure 4: Edit/Create labyrinth menu: here you can create any size of labyrinths with any number of energy sources or sinks. Clicking to *Load existing lab* button to import an existing labyrinth

SQLite database file.

Click to *Create Labyrinth* green button to create a new labyrinth or import existing ones (Fig. 4). If you want to import an existing labyrinth to use the same labyrinth as previous simulations click on the *Load existing lab* button. OK is the accept button.

Click to *Create Workers* green button to create a new workers (Fig.5). OK is the accept button. There are checkboxes where you can set learning and merging options. If neither was checked workers do not learn, they walk randomly in the labyrinth.

If you click on *Set parameters* button you can set up parameters (Figure 6) of the world: initial worker energy, movement costs, source and sink energy max, replication energy level, replication rate (default is 2), delay. Source and sink energy are random numbers between 0 and '...max'. If a worker's energy exceeds or is over the replication energy level it can make offspring (default value is 2). Delay is a labyrinth parameter. If a source has been

Create workers

Set up workers

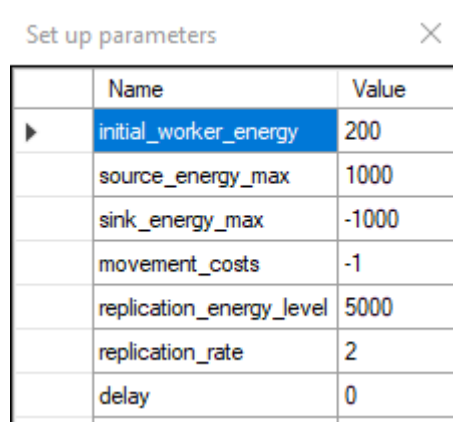
Number of workers:

Create 200 workers

|   | id | Energy | Parent |
|---|----|--------|--------|
| ▶ | 0  | 200    |        |
|   | 1  | 200    |        |
|   | 2  | 200    |        |
|   | 3  | 200    |        |
|   | 4  | 200    |        |
|   | 5  | 200    |        |
|   | 6  | 200    |        |
|   | 7  | 200    |        |
|   | 8  | 200    |        |
|   | 9  | 200    |        |
|   | 10 | 200    |        |
|   | 11 | 200    |        |
|   | 12 | 200    |        |
|   | 13 | 200    |        |

Close

Figure 5: Here you can create workers. The number of workers has to be set up before clicking to *Create workers* button.



The image shows a window titled "Set up parameters" with a close button (X) in the top right corner. Inside the window is a table with two columns: "Name" and "Value". The first row is highlighted in blue. The parameters listed are: initial\_worker\_energy (200), source\_energy\_max (1000), sink\_energy\_max (-1000), movement\_costs (-1), replication\_energy\_level (5000), replication\_rate (2), and delay (0).

|   | Name                     | Value |
|---|--------------------------|-------|
| ▶ | initial_worker_energy    | 200   |
|   | source_energy_max        | 1000  |
|   | sink_energy_max          | -1000 |
|   | movement_costs           | -1    |
|   | replication_energy_level | 5000  |
|   | replication_rate         | 2     |
|   | delay                    | 0     |

Figure 6: The setup parameters purpose is to adjust the circumstances in the labyrinths. You can edit world parameters if you change any values. The changed value is stored if you select the next record.

eaten this source becomes disabled, and unreachable for one or two iteration cycles depending on the delay value.

If you have set up the necessary parameters click to *Start iteration* green button. The simulation process iterates until the *Stop iteration* red button is clicked or the iteration count exceeds the limit (Stop iteration when step count is ...) or all workers have died.

If you want to save all produced data during the simulation set the checkbox Save data to DB file to true and give the starting character of the SQLite file name. Further elements of this file name are generated by the simulator from labyrinth size, workers count, learning options, start position, etc.

Data are stored in an SQLite database file with tables iteration, workers, and labyrinth. These are for use by the DCAnalyser later on.

Every simulation has a report file where the main metadata are stored after the iteration is terminated. This is a text file with the extension '.rep'. The iteration steps are also stored in a text file with the extension '.iter'. This file is needed for the 'on-the-fly analyser' built in the DCMaster.

Every stored data can be found in the 'Special Directories' of Windows for example C:\Users\username\AppData\Local\DC. DC has the following subfolders: iteration, labyrinths, databases, reports, figures (if you made screenshots).



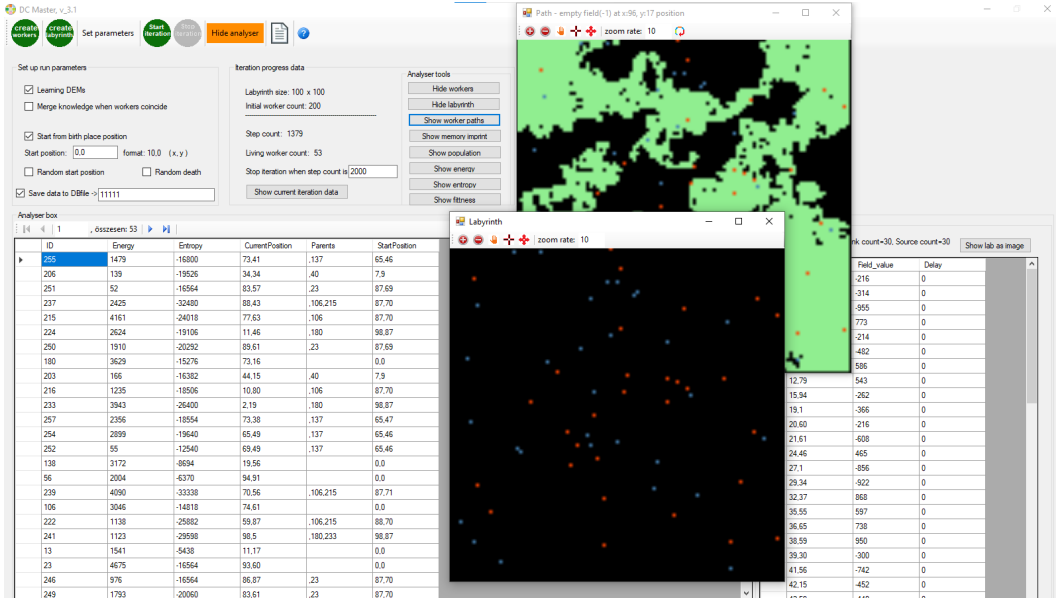


Figure 7: On-the-fly analyser's complex display

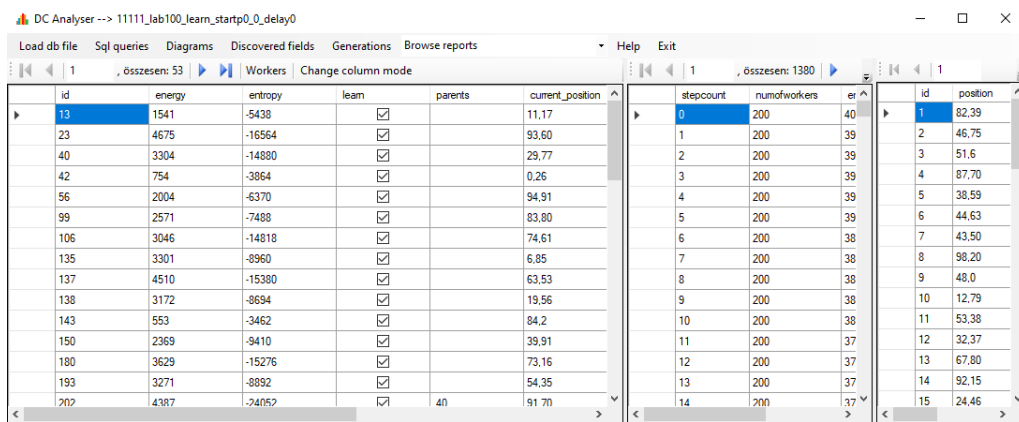
## 5.1 Built-in analyser

There is a *Show analyser* button in the DCMaster form. Use this if you want to analyse the iteration on-the-fly. When you click *Show analyser* button the simulator form switches to full-screen. Many new buttons appear and we can display the population, worker path, energy, entropy, and fitness in diagrams, but you can see worker data in a data grid, the labyrinth in numerical or image form (Fig.7).

The capabilities of the on-the-fly analyser is much more limited than the DCAnalyser, but during long iterations can be very useful.

## 6 DCAnalyser

If you already have simulation results you can use DCAnalyser.exe to analyse iteration databases to observe events, to understand what happened to evolutionary entities, and what the fate of DEM workers is. There are built-in queries, charts, and picture-generation possibilities to help your understanding. You can create specific queries too (*Sql* menu), and you can display results graphically (*Diagrams* menu). In *Selection* menu you can see discovered fields of a labyrinth. Figure 8 demonstrates the start form of DCAnalyser. You can see a labyrinth not only in tabular form but in graphics too if you



| id  | energy | entropy | learn                               | parents | current_position |
|-----|--------|---------|-------------------------------------|---------|------------------|
| 13  | 1541   | -5438   | <input checked="" type="checkbox"/> |         | 11,17            |
| 23  | 4675   | -16564  | <input checked="" type="checkbox"/> |         | 93,60            |
| 40  | 3304   | -14880  | <input checked="" type="checkbox"/> |         | 29,77            |
| 42  | 754    | -3864   | <input checked="" type="checkbox"/> |         | 0,26             |
| 56  | 2004   | -6370   | <input checked="" type="checkbox"/> |         | 94,91            |
| 99  | 2571   | -7488   | <input checked="" type="checkbox"/> |         | 83,80            |
| 106 | 3046   | -14818  | <input checked="" type="checkbox"/> |         | 74,61            |
| 135 | 3301   | -8960   | <input checked="" type="checkbox"/> |         | 6,85             |
| 137 | 4510   | -15380  | <input checked="" type="checkbox"/> |         | 63,53            |
| 138 | 3172   | -8694   | <input checked="" type="checkbox"/> |         | 19,56            |
| 143 | 553    | -3462   | <input checked="" type="checkbox"/> |         | 84,2             |
| 150 | 2369   | -9410   | <input checked="" type="checkbox"/> |         | 39,91            |
| 180 | 3629   | -15276  | <input checked="" type="checkbox"/> |         | 73,16            |
| 193 | 3271   | -8892   | <input checked="" type="checkbox"/> |         | 54,35            |
| 202 | 4387   | -74152  | <input checked="" type="checkbox"/> | 40      | 91,70            |

| stepcount | numofworkers | energy |
|-----------|--------------|--------|
| 0         | 200          | 40     |
| 1         | 200          | 39     |
| 2         | 200          | 39     |
| 3         | 200          | 39     |
| 4         | 200          | 39     |
| 5         | 200          | 39     |
| 6         | 200          | 38     |
| 7         | 200          | 38     |
| 8         | 200          | 38     |
| 9         | 200          | 38     |
| 10        | 200          | 38     |
| 11        | 200          | 37     |
| 12        | 200          | 37     |
| 13        | 200          | 37     |
| 14        | 200          | 37     |

| id | position |
|----|----------|
| 1  | 82,39    |
| 2  | 46,75    |
| 3  | 51,6     |
| 4  | 87,70    |
| 5  | 38,59    |
| 6  | 44,63    |
| 7  | 43,50    |
| 8  | 98,20    |
| 9  | 48,0     |
| 10 | 12,79    |
| 11 | 53,38    |
| 12 | 32,37    |
| 13 | 67,80    |
| 14 | 92,15    |
| 15 | 24,46    |

Figure 8: DCAnalyser: This module has wide range functions of data analysis. You can see data in tabular form, and graphics too, such as charts or images

click on the labyrinth data grid with the right mouse button. You can see workers in a separate data grid, and the iteration steps in another data grid.

## 6.1 Load Db file menu

After DCAnalyser.exe starts you can analyse the database. First, click on *Load DB file* menu item to analyse events and data. When a database has been selected its content has loaded to different tables (workers, iterations, labyrinths) as you can see in the figure (Fig.8)

If the selected database has been loaded functionalities become reachable. For example, by right-mouse clicking anywhere on the labyrinth data grid it can be seen in graphic format as a bitmap (Fig.9). If you move the cursor on the bitmap the field type (source or sink) and energy content will appear in the header of the form.

## 6.2 Sql menu

You can make any selections with *Sql menu* (Fig.12). If you know Sql you can create an Sql query, and you can see it in tabular form. If you choose *Display charts from any Select* sub menu you can see the result in a chart form (Fig.13).

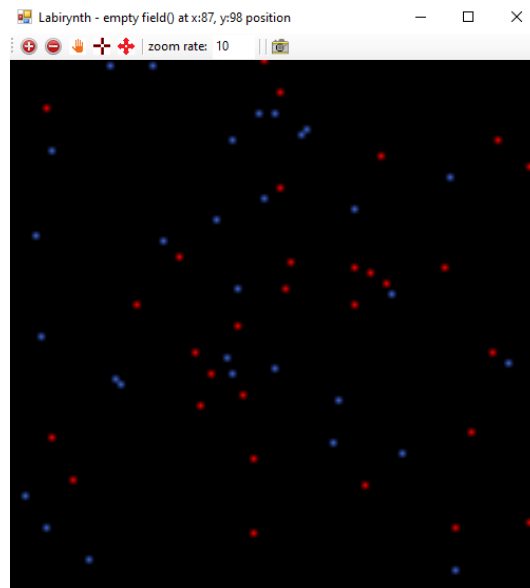


Figure 9: This is the labyrinth with energy sources (red spots) and sinks (blue spots)

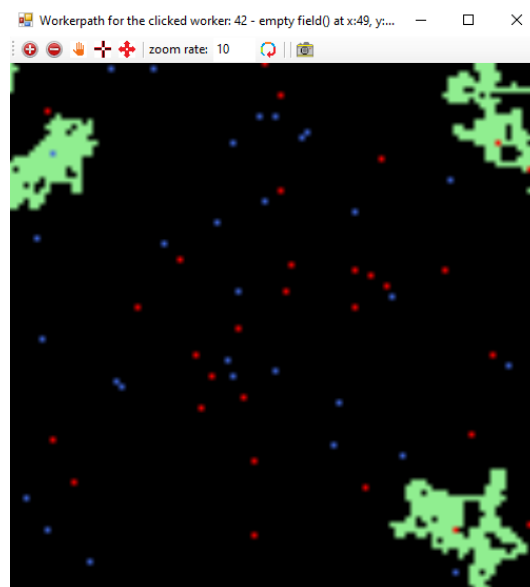


Figure 10: This is the worker path of the selected worker. Green pixels mean the reached fields

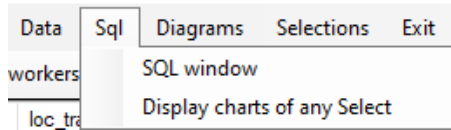


Figure 11: With Sql menu you can create arbitrary Sql selections. You can display results in tabular or graphic form

SQL window

Sql command --> select \* from workers where parents=""

Load saved Sql

Save command

Previous SQL commands:

összesen: 33

|   | id  | energy | entropy | learn                               | parents | current_position | start_pos |
|---|-----|--------|---------|-------------------------------------|---------|------------------|-----------|
| ▶ | 12  | 4298   | -18690  | <input checked="" type="checkbox"/> |         | 35,182           | 50,0      |
|   | 15  | 947    | -4818   | <input checked="" type="checkbox"/> |         | 21,171           | 50,0      |
|   | 24  | 1086   | -4254   | <input checked="" type="checkbox"/> |         | 42,151           | 50,0      |
|   | 27  | 3837   | -12954  | <input checked="" type="checkbox"/> |         | 66,20            | 50,0      |
|   | 29  | 4214   | -17720  | <input checked="" type="checkbox"/> |         | 26,178           | 50,0      |
|   | 33  | 4736   | -11536  | <input checked="" type="checkbox"/> |         | 46,192           | 50,0      |
|   | 41  | 3980   | -17190  | <input checked="" type="checkbox"/> |         | 33,6             | 50,0      |
|   | 45  | 1098   | -4254   | <input checked="" type="checkbox"/> |         | 39,30            | 50,0      |
|   | 61  | 1829   | -5746   | <input checked="" type="checkbox"/> |         | 28,0             | 50,0      |
|   | 63  | 1129   | -4318   | <input checked="" type="checkbox"/> |         | 18,46            | 50,0      |
|   | 65  | 1914   | -5864   | <input checked="" type="checkbox"/> |         | 23,181           | 50,0      |
|   | 70  | 2785   | -9892   | <input checked="" type="checkbox"/> |         | 54,167           | 50,0      |
|   | 77  | 3661   | -14180  | <input checked="" type="checkbox"/> |         | 57,4             | 50,0      |
|   | 83  | 1111   | -4254   | <input checked="" type="checkbox"/> |         | 70,186           | 50,0      |
|   | 91  | 3631   | -10118  | <input checked="" type="checkbox"/> |         | 62,16            | 50,0      |
|   | 92  | 2689   | -7410   | <input checked="" type="checkbox"/> |         | 44,34            | 50,0      |
|   | 111 | 3852   | -12038  | <input checked="" type="checkbox"/> |         | 195,148          | 50,0      |

Figure 12: Sql window has an SQL command line where you can type your Sql command. After successful execution the result appears in a data grid

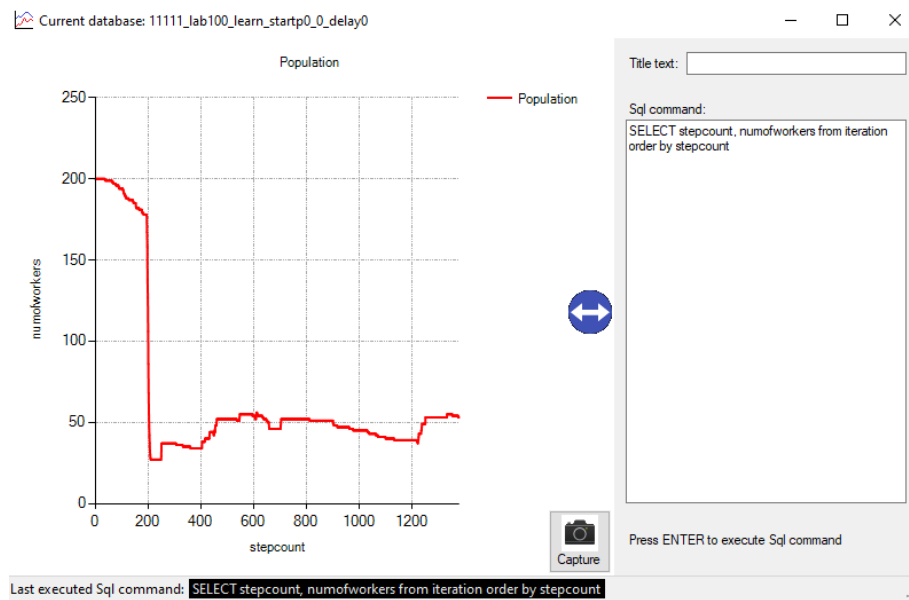


Figure 13: This is Sql window with a chart. In this case we selected the iteration count versus population.

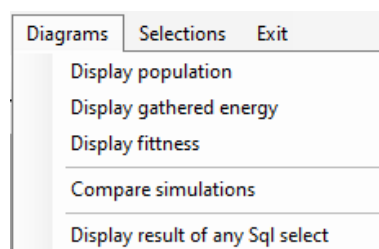


Figure 14: You can draw charts if you choose any sub menus from *Th Diagram* menu

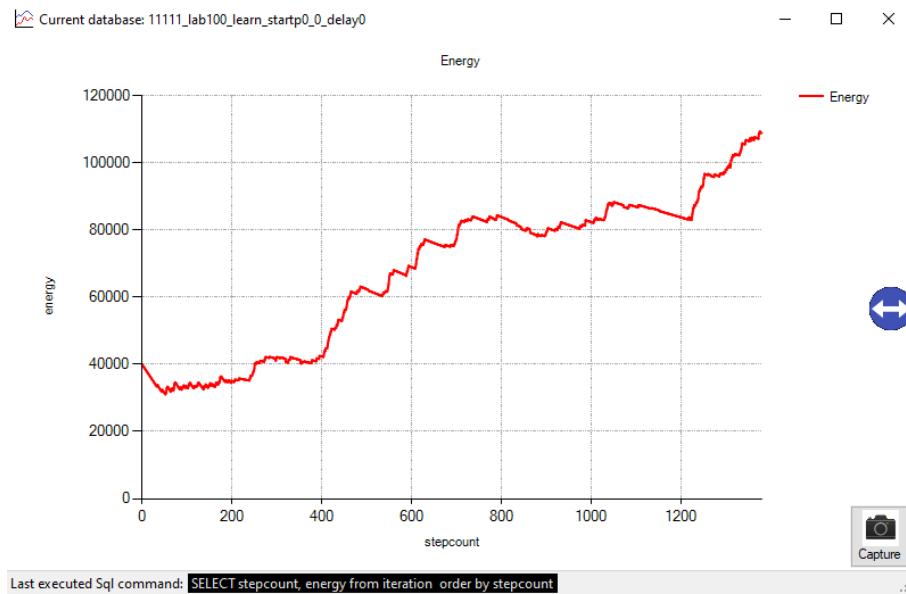


Figure 15: In this case, we selected the Energy submenu to display the gathered energy in a chart

### 6.3 Diagram menu

In *Diagram* menu (Fig.14) you can see simulation data on a chart: population (number of alive workers), population rate, gathered energy, energy rate, and fitness of groups or imprint (known energy sources and sinks). In Fig.15 the gathered energy of the existing population can be seen. If you click on the Capture icon this graphics has been saved in a bitmap file in the `...\DC\figures\` folder.

You can also compare any data from any simulation. If you click on the special icon (white arrows) an SQL command field displays, where you can see the actual Sql statement.

You can not only display charts on a certain database but compare data of different simulations (Fig.16). If you choose *Compare simulations* menu you can choose the desired simulation database name and data type to display.

### 6.4 Discovered fields menu

In the *Discovered fields* menu we can see paths in an image where workers have visited. If you select one worker you can see its worker path, if not you can see the worker path for all living workers.

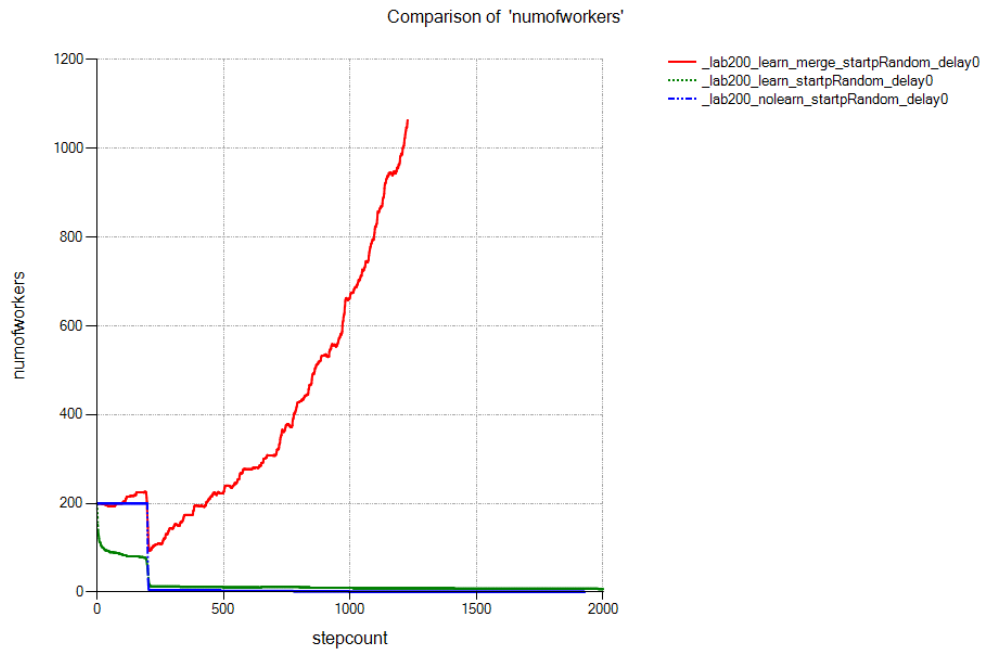


Figure 16: Comparison the populations of two different simulations

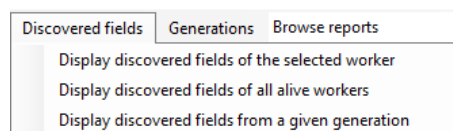


Figure 17: The *Discoveredfields* menu with submenus, where you can see the paths of all alive workers or a selected worker's one.

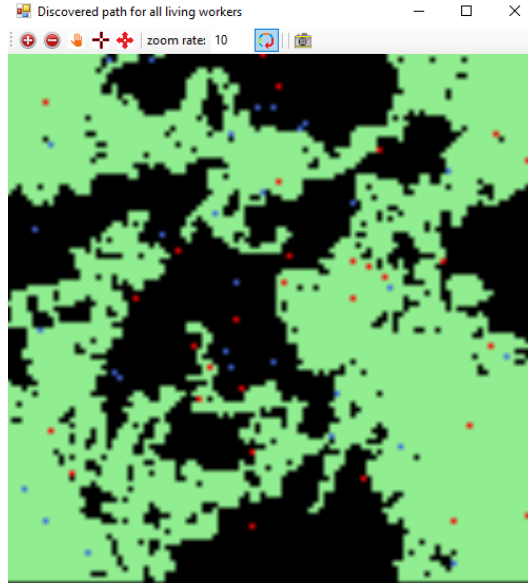


Figure 18: This figure shows workers paths for all living workers. Green pixels symbolize the visited fields

## 6.5 Generations menu

Understanding the fates of generations can be interesting. Since almost every initial worker has died at some first iteration step thus especially interesting what happened to those who did not extinct. That's why were worked out the Generations menu.

Do not forget the family tree can be huge since the population can be multiplied to more thousands. Thus it is impossible to display the whole family tree on the screen. We have more ways to display the family tree. The following figures (Fig.19, 20, 21) illustrate these possibilities.

In this menu, you can select initial workers, if they exist at all, select different generations, one worker's parents, and all living workers.



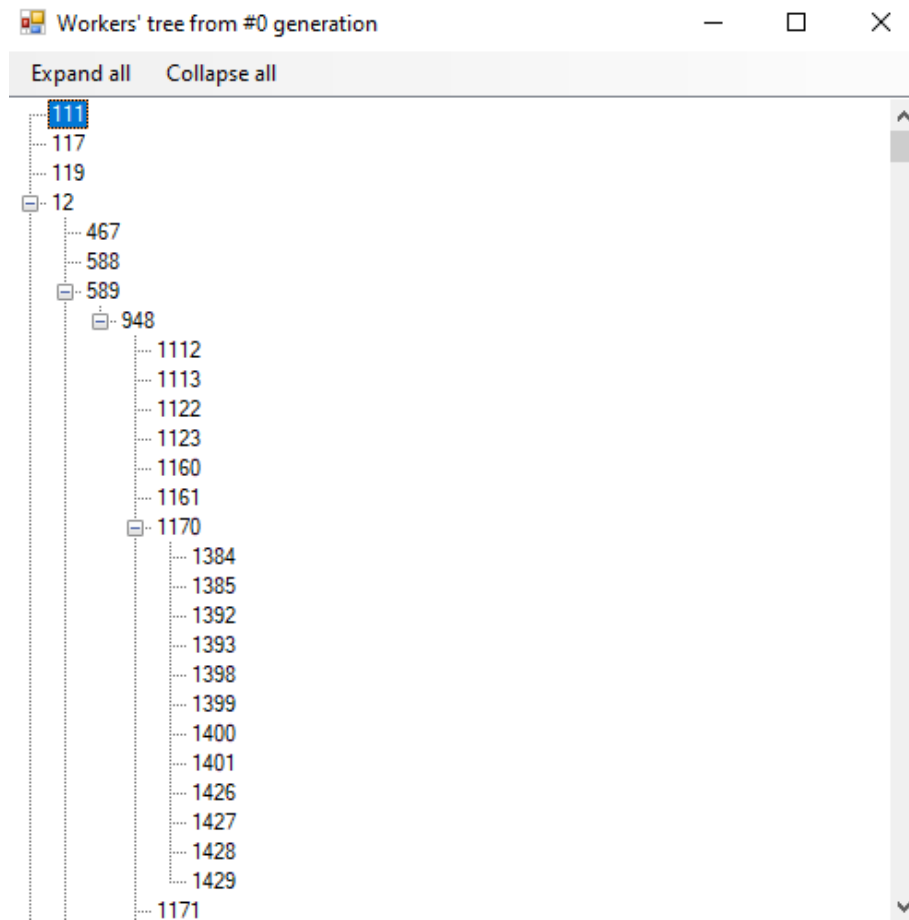


Figure 19: This is a traditional family tree. You can scroll along all workers where you can see parents, grandparents, aunts, and uncles

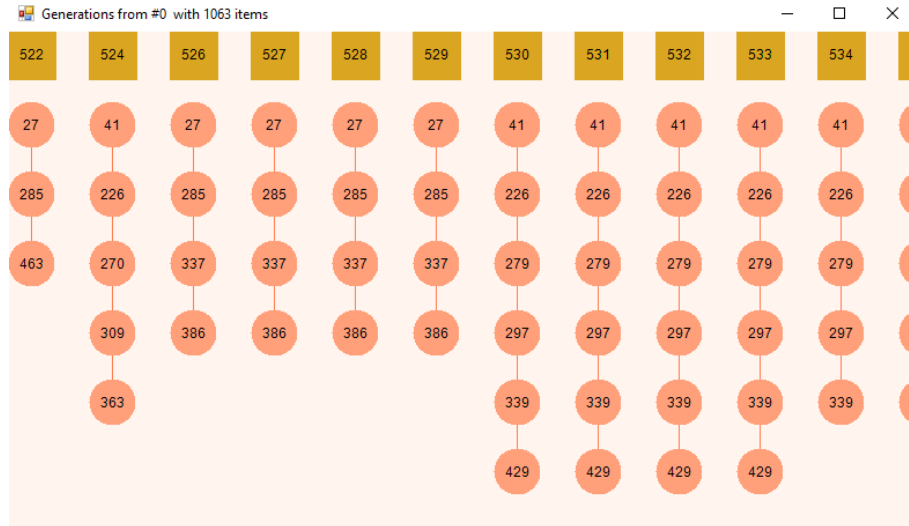


Figure 20: This is another style of graphical description of the family tree. The living workers are on the upper side of the form in goldenrod colour boxes with their IDs. The orange colour circles are the parents with their IDs

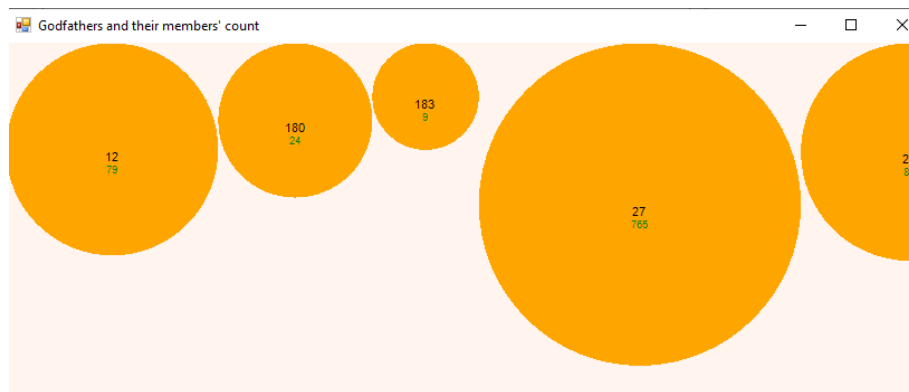


Figure 21: This figure displays clans with their IDs and the count of closer and remoter members of their families