DEM Center Users' Guide

Istvan Elek

December 6, 2018

Contents

1 Preface

DEM center 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 the ziped (dc.zip) package from this site (Download menu)
2. Copy it to the desired directory and unzip
3. Download Postgres 9.6 or later version and install it
4. Download and install pgAdmin 4.x (optional)
5. Create a postgres user with name: wumpus with password: wumpus with admin rights (use postgres command line or pgAdmin). If you want to use your own parameters, open *config.cfg* file, from the program directory and put your data into it (username: wumpus, password: wumpus, host: localhost). You may also need to change the content of *configAdmin.cfg* file (username postgres, password root, database postgres, host localhost) if your data are different from them.
6. Start DC.exe
7. Let's start!

3 DC

DC.exe is a frame program, which organize your work. You can make the followings (Figure 1):

- You can start DCDemo.exe, which demonstrates graphically the movements of DEM entities (workers) in small labyrinth. There is no database connection and knowledge base creation in this case.

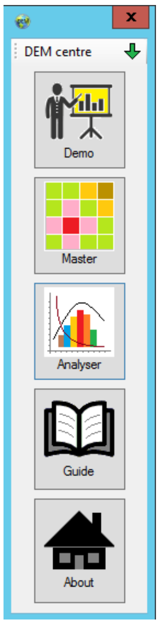


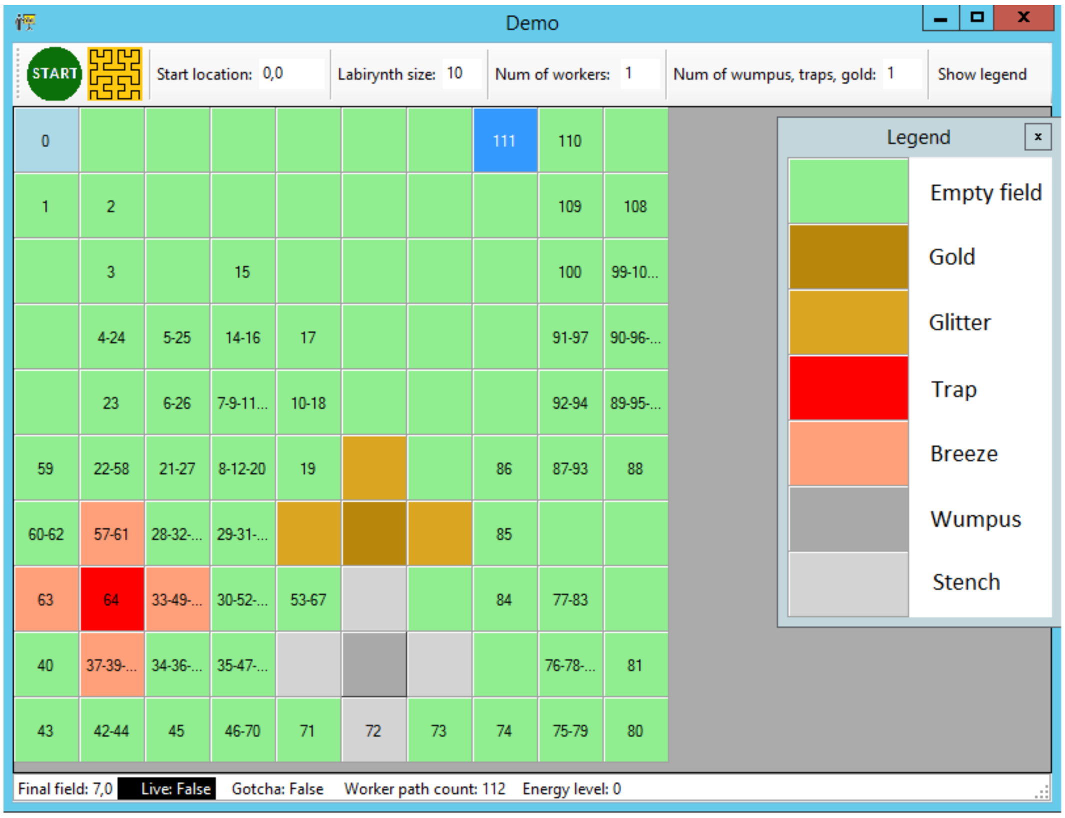
Figure 1: DC Centre form: Demo, Master, Analyser modules can be launched from here.

- 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 have 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.

4 DCDemo

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



Start location: 0,0 Labirynth size: 10 Num of workers: 1 Num of wumpus, traps, gold: 1 Show legend

0							111	110	
1	2							109	108
	3		15					100	99-10...
	4-24	5-25	14-16	17				91-97	90-96-...
	23	6-26	7-9-11...	10-18				92-94	89-95-...
59	22-58	21-27	8-12-20	19			86	87-93	88
60-62	57-61	28-32-...	29-31-...				85		
63	64	33-49-...	30-52-...	53-67			84	77-83	
40	37-39-...	34-36-...	35-47-...					76-78-...	81
43	42-44	45	46-70	71	72	73	74	75-79	80

Legend

🟩	Empty field
🟫	Gold
🟨	Glitter
🟥	Trap
🟧	Breeze
⬛	Wumpus
⬜	Stench

Final field: 7,0 Live: False Gotcha: False Worker path count: 112 Energy level: 0

Figure 2: DCDemo: Labyrinth with legend and movements can be seen. Numbers symbolize the step number when the certain field was entered.

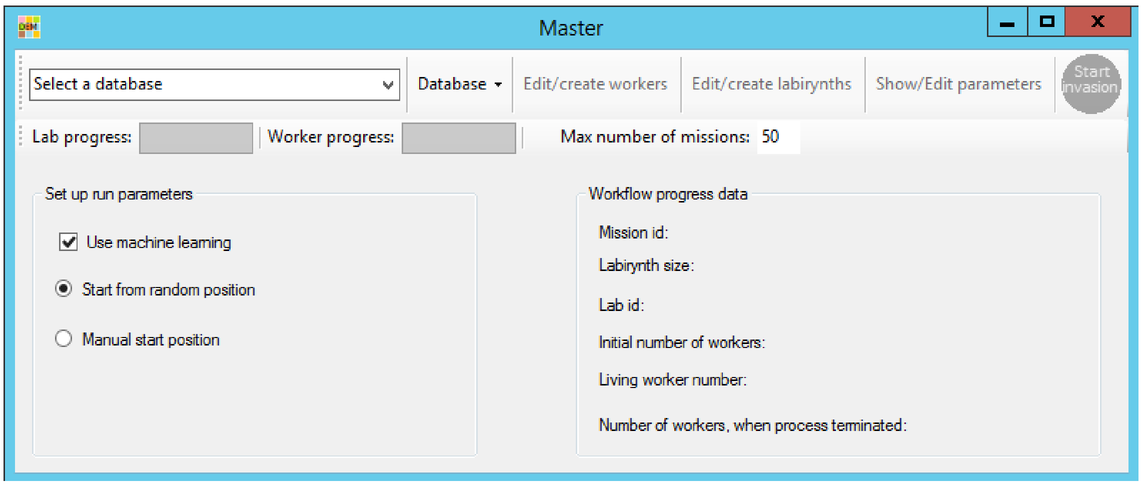


Figure 3: The DCmaster starter screen: This is where you can open an existing database (invasion) or create any labyrinths, DEM workers, and you can set up simulation parameters.

5 DCMaster

If you are going to make simulations, start DCMaster.exe (Figure 3). First, you need to create a new database (invasion) in the *Database* menu (Figure 4). The database name must be started with 'dem' letters. Also in it you can edit/display config file (Figure 5), which contains connection parameters to Postgres and existing dem databases.

After creation you should select this new database. Click to 'Select a database' combo box to select the desired database. If you created a new database (invasion), or you selected an existing one, you can create DEM workers by clicking *Edit/Create workers* menu (Figure 6).

You can also create any number and size of labyrinths by clicking *Edit/Create labyrinths* menu (Figure 7), or edit existing one. You can set the number and size of labyrinths, and the number of wumpuses, traps and gold bars.

You can chose operation modes too, such as use machine learning, or not use. You can set up starting position by hand or by random method. After you clicked to *StartInvasion* button you can see summary data after every closed session in 'Workflow progress data' section.

You can set up parameters (Figure 8) of the world: initial worker energy, movement costs, wumpus, trap, gold energy content, replication energy level, replication rate (default is 2), etc.

In the *Database* menu there are three further sub menus. The first is

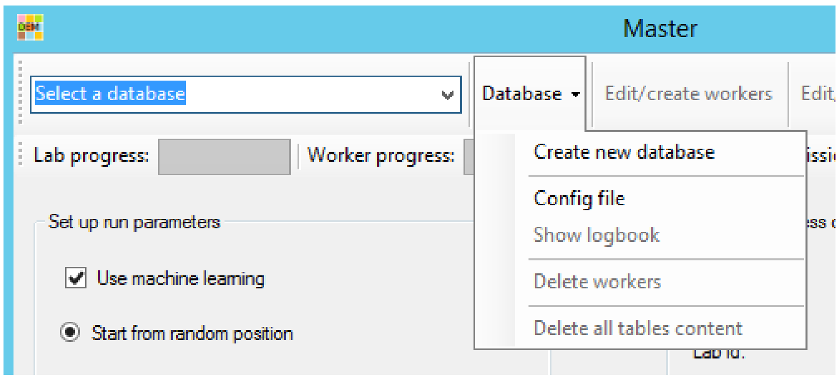
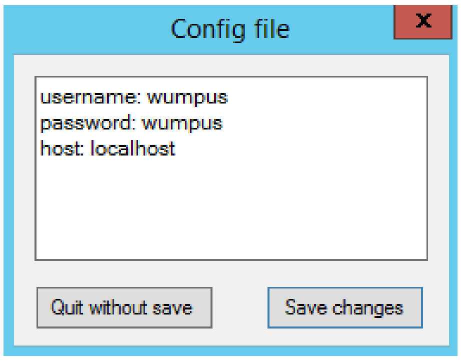


Figure 4: Database menu, where you can create new database, or edit config file



```
username: wumpus
password: wumpus
host: localhost
```

Figure 5: Database/Config file menu, where you can edit the config file content

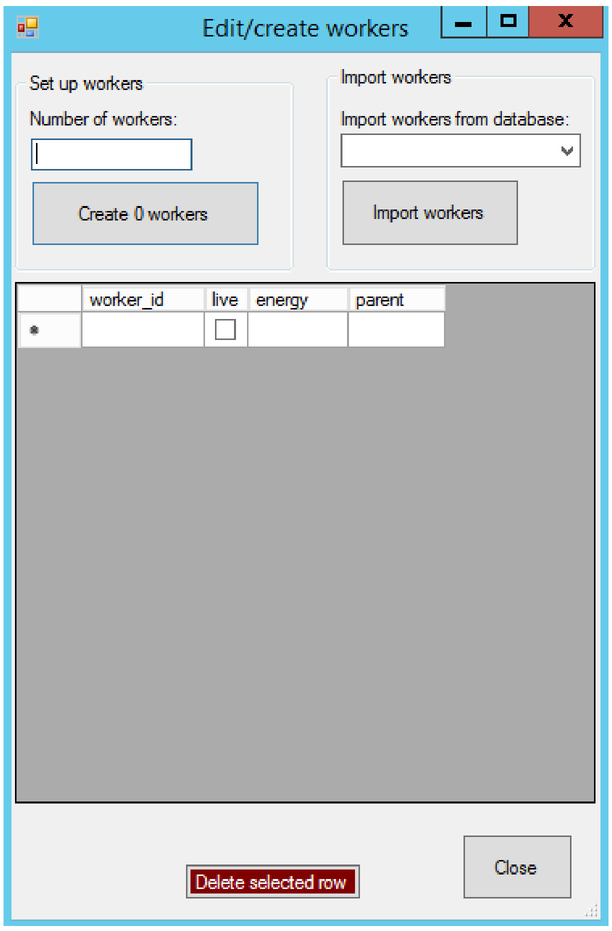
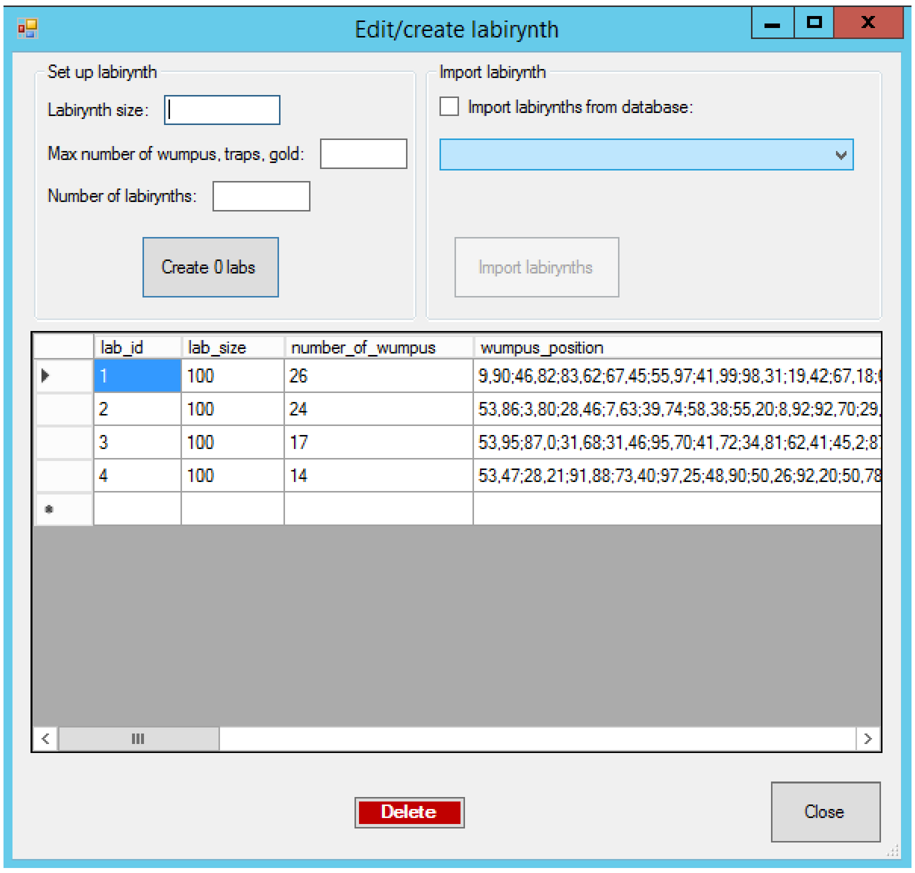


Figure 6: Edit/Create worker menu: here you can create or delete the selected workers. The number of workers has to be set up before clicking to 'Create workers' button.,



Labirynth size:

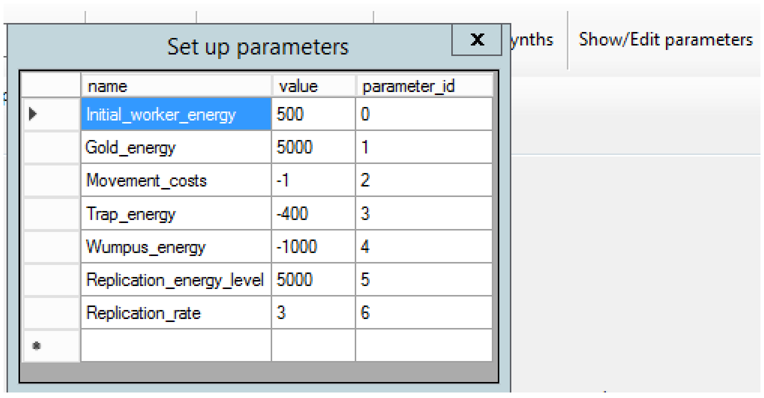
Max number of wumpus, traps, gold:

Number of labirynths:

Import labirynths from database:

	lab_id	lab_size	number_of_wumpus	wumpus_position
▶	1	100	26	9,90;46,82;83,62;67,45;55,97;41,99;98,31;19,42;67,18;
	2	100	24	53,86;3,80;28,46;7,63;39,74;58,38;55,20;8,92;92,70;29,
	3	100	17	53,95;87,0;31,68;31,46;95,70;41,72;34,81;62,41;45,2;8
	4	100	14	53,47;28,21;91,88;73,40;97,25;48,90;50,26;92,20;50,78
*				

Figure 7: Edit/Create labyrinth menu: here you can create any number and size of labyrinths with any number of wumpuses, traps and gold bars. You can also delete a selected labyrinth.


Set up parameters

	name	value	parameter_id
▶	Initial_worker_energy	500	0
	Gold_energy	5000	1
	Movement_costs	-1	2
	Trap_energy	-400	3
	Wumpus_energy	-1000	4
	Replication_energy_level	5000	5
	Replication_rate	3	6
*			

Figure 8: Set up parameters menu is for adjusting the circumstances in the artificial world (labyrinths). You can edit world parameters if you change values. The changed value is stored if you select the next record.

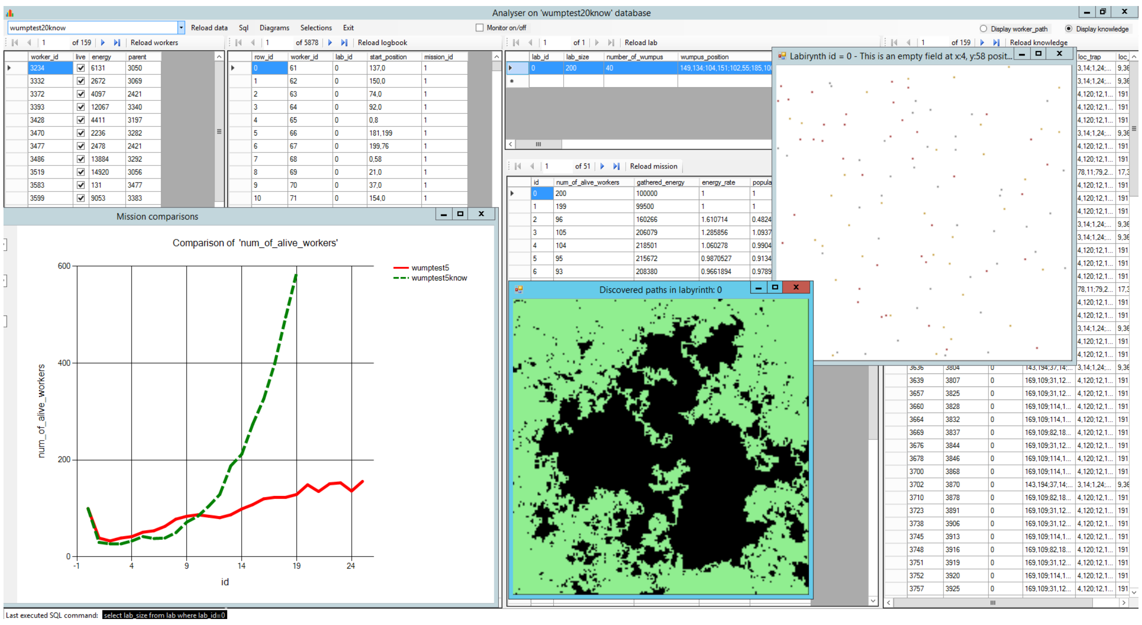
Delete workers, which delete all existing workers from the invasion database and worker related table such as worker_path and knowledge. It is useful if you wish to start a new invasion with new workers, but in old labyrinths. To click to *Delete all tables content* menu item deletes every table from the selected (actual) database such as workers, labyrinths, logbook, worker_path and knowledge. Finally, you can display logbook content by clicking the *Show logbook* menu.

If every necessary parameter has set up, and labs and workers have been created you can start the simulation by clicking to *Start invasion* button. You can observe events in 'Workflow progress data' section. The better mode of observation is to start DCAnalyser.exe which has many functions to display and analyse data.

There is a report file (invasion_database_name.rep) which is a summary of an invasion stored in a text file in the /user/documents/DCreports/ directory.

6 DCAnalyser

If you already have simulation results, or you have just started the simulation process by clicking *Start invasion* button, you can use DCAnalyser.exe to analyse databases (invasions data), to understand what happened to evolutionary entities, and what the fates of DEM workers are. There are built-in queries, charts and picture to help your understanding. You can create your own selections too (*Sql* menu), and you can display results (*Diagrams* menu). In *Selection* menu you can see knowledge data in tabular form, or you can



worker_id	live	energy	parent
3234	✓	6131	3050
3332	✓	2672	3069
3372	✓	4097	2421
3393	✓	12067	3340
3428	✓	4411	3197
3470	✓	2236	3282
3477	✓	2478	2421
3486	✓	13884	3292
3519	✓	14920	3056
3583	✓	131	3477
3599	✓	9053	3383

row_id	worker_id	lab_id	start_position	mission_id
0	61	0	137,0	1
1	62	0	150,0	1
2	63	0	74,0	1
3	64	0	92,0	1
4	65	0	0,8	1
5	66	0	181,199	1
6	67	0	199,76	1
7	68	0	0,58	1
8	69	0	21,0	1
9	70	0	37,0	1
10	71	0	154,0	1

lab_id	lab_size	number_of_wumpus	wumpus_position
0	200	40	149,134;104,151;102,55;185,10

id	num_of_alive_workers	gathered_energy	energy_rate	popula
0	200	100000	1	1
1	199	99500	1	1
2	96	160266	1.610714	0.4824
3	105	206079	1.285856	1.0937
4	104	218501	1.060278	0.9904
5	95	215672	0.9870527	0.9134
6	93	208380	0.9661894	0.9789

				loc_trap	loc_
				3,14;1,24;...	9,36
				3,14;1,24;...	9,36
				4,120;12,1...	191
				4,120;12,1...	191
				4,120;12,1...	191
				3,14;1,24;...	9,36
				4,120;12,1...	191
				4,120;12,1...	191
				78,11;79,2...	17,3
				4,120;12,1...	191
				4,120;12,1...	191
				4,120;12,1...	191
				3,14;1,24;...	9,36
				3,14;1,24;...	9,36
				4,120;12,1...	191
				4,120;12,1...	191
				4,120;12,1...	191
				78,11;79,2...	17,3
				4,120;12,1...	191
				4,120;12,1...	191
				3,14;1,24;...	9,36
				4,120;12,1...	191
				4,120;12,1...	191
3636	3804	0	143,194;37,14;...	3,14;1,24;...	9,36
3639	3807	0	169,109;31,12...	4,120;12,1...	191
3657	3825	0	169,109;31,12...	4,120;12,1...	191
3660	3828	0	169,109;114,1...	4,120;12,1...	191
3664	3832	0	169,109;114,1...	4,120;12,1...	191
3669	3837	0	169,109;82,18...	4,120;12,1...	191
3676	3844	0	169,109;31,12...	4,120;12,1...	191
3678	3846	0	169,109;114,1...	4,120;12,1...	191
3700	3868	0	169,109;114,1...	4,120;12,1...	191
3702	3870	0	143,194;37,14;...	3,14;1,24;...	9,36
3710	3878	0	169,109;82,18...	4,120;12,1...	191
3723	3891	0	169,109;31,12...	4,120;12,1...	191
3738	3906	0	169,109;31,12...	4,120;12,1...	191
3745	3913	0	169,109;114,1...	4,120;12,1...	191
3748	3916	0	169,109;82,18...	4,120;12,1...	191
3751	3919	0	169,109;31,12...	4,120;12,1...	191
3752	3920	0	169,109;114,1...	4,120;12,1...	191
3757	3925	0	169,109;31,12...	4,120;12,1...	191

Figure 9: 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

see discovered fields of a labyrinth. Figure 9 demonstrates the complexity of DCAnalyser. You can see a labyrinth not only in tabular form, but in graphics too if you click to a labyrinth with right mouse button.

After DCAnaliser.exe start you need to analyse the database. Select a database (invasion) to analyse events and data (Figure 10). When a database has been selected its content were loaded to different tables (workers, logbook, labyrinths, worker-Path or knowledge). If you checked 'Display knowledge' (Figure 11), you will see workers' knowledge in the datagridview. If you checked 'Display worker path' you will see the the worker path of all workers.

The selected database content should be reloaded (Figure 12), if the content of selected database is changing (for example a simulation is running just now). If you use a monitor option (check it on) in main menu bar you can observe the number of alive workers in real time.

You can make any selections with *Sql* menu (Figure 13). if you know Sql you can create your own 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 chart form.

In *Diagram* menu (Figure 14) you can see mission data on charts: pop-

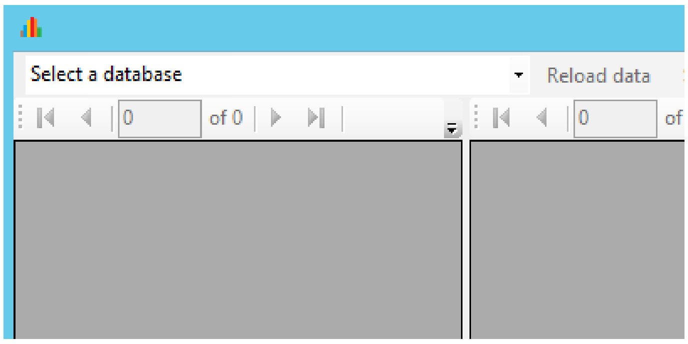


Figure 10: Click to 'Select database' combo to select a database which contains every data of the selected invasion

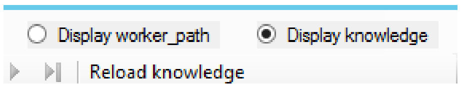


Figure 11: If you check *Display knowledge* (default) you can see the knowledge table content. If *Display worker path* is checked you can see the worker_path table contents (be careful, because this table can be enormous with million records)

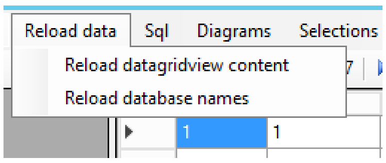


Figure 12: Reload menu: You can reload the selected database content. You can also reload database names. It is needed if you started a new simulation with DCMaster.exe, and a new simulation database has been created.

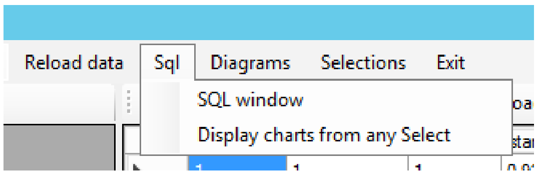


Figure 13: With Sql menu you can create arbitrary Sql selections. You can display results in tabular or graphic form

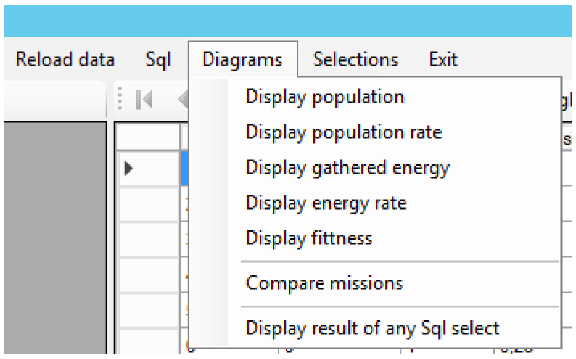


Figure 14: You can draw charts if you choose any sub menus from *Diagram* menu.

ulation (number of alive workers), population rate, gathered energy, energy rate, and fitness of groups. You can also compare any data from any mission. If you click to the special icon (white arrows) an sql command field displays, where you can see actual Sql statement.

You can not only display charts on the certain database (invasion), but compare mission data of different invasions (Figure 15). If you choose *Compare missions* menu you can choose desired invasions and data type to display.

Choose *Selections* menu (Figure 16) if you wish to see workers' knowledge for selected labyrinth or for all labyrinth. If the workers knowledge is interesting, first you should select a certain worker by clicking to a record in'workers' table. If you did not select labyrinth the selected knowledge is valid for all labyrinths.

In *Selections* menu you can display the discovered fields of a selected labyrinth (Figure 17). You can see individual or collective knowledge as an image. The union of individual results is the collective knowledge.

You can see a labyrinth graphically if you make a mouse click with right mouse button to a record in lab table data grid view. I this was you will see the distribution of objects in the selected labyrinth (Figure 18).

Clicking to the menu item 'Display discovered fields of a given generation' we you query generations' prosperity. You can set up generation number (1,2,...) where 1 means the first generation (there is no ancestors of them) and quantifier ($<, >, =$, etc). The result is displayed graphically (visited point of the selected labyrinth) and optionally, you can see it in tabular form.

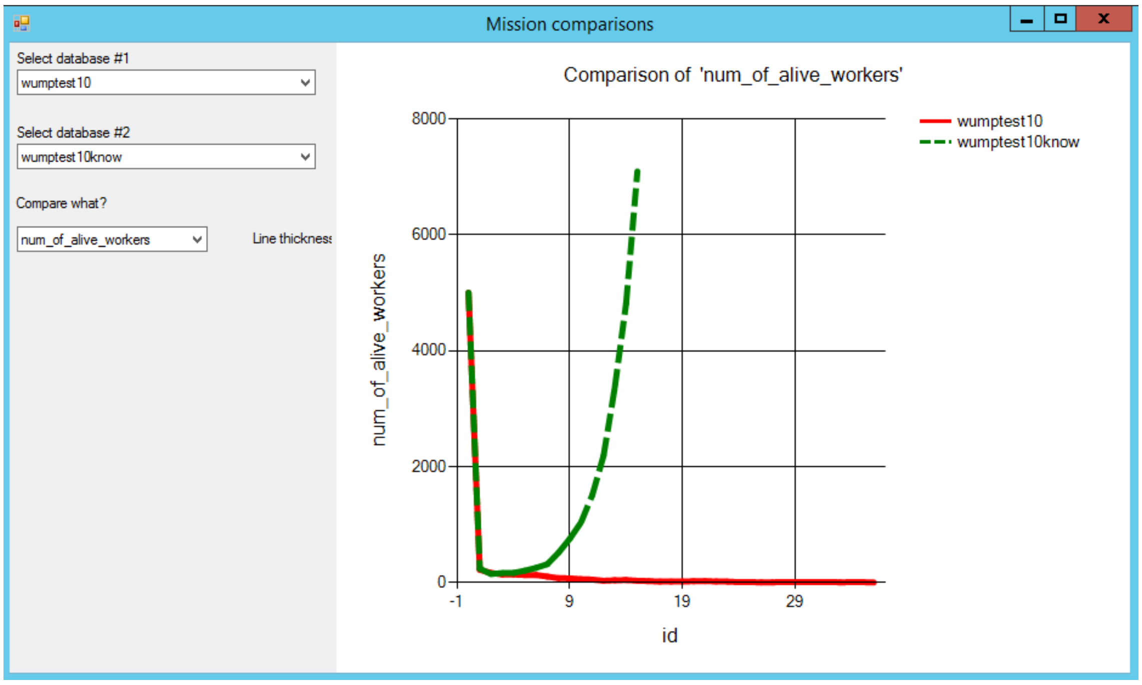


Figure 15: Comparison of two invasions database.

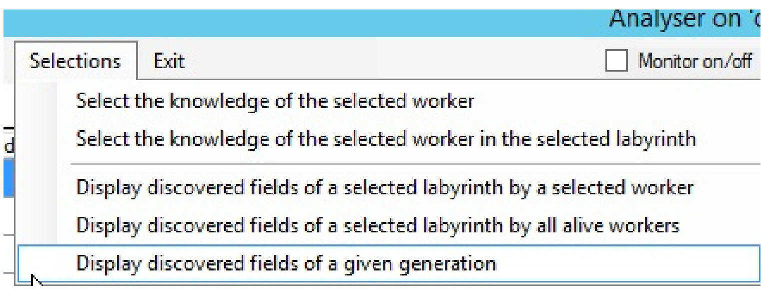
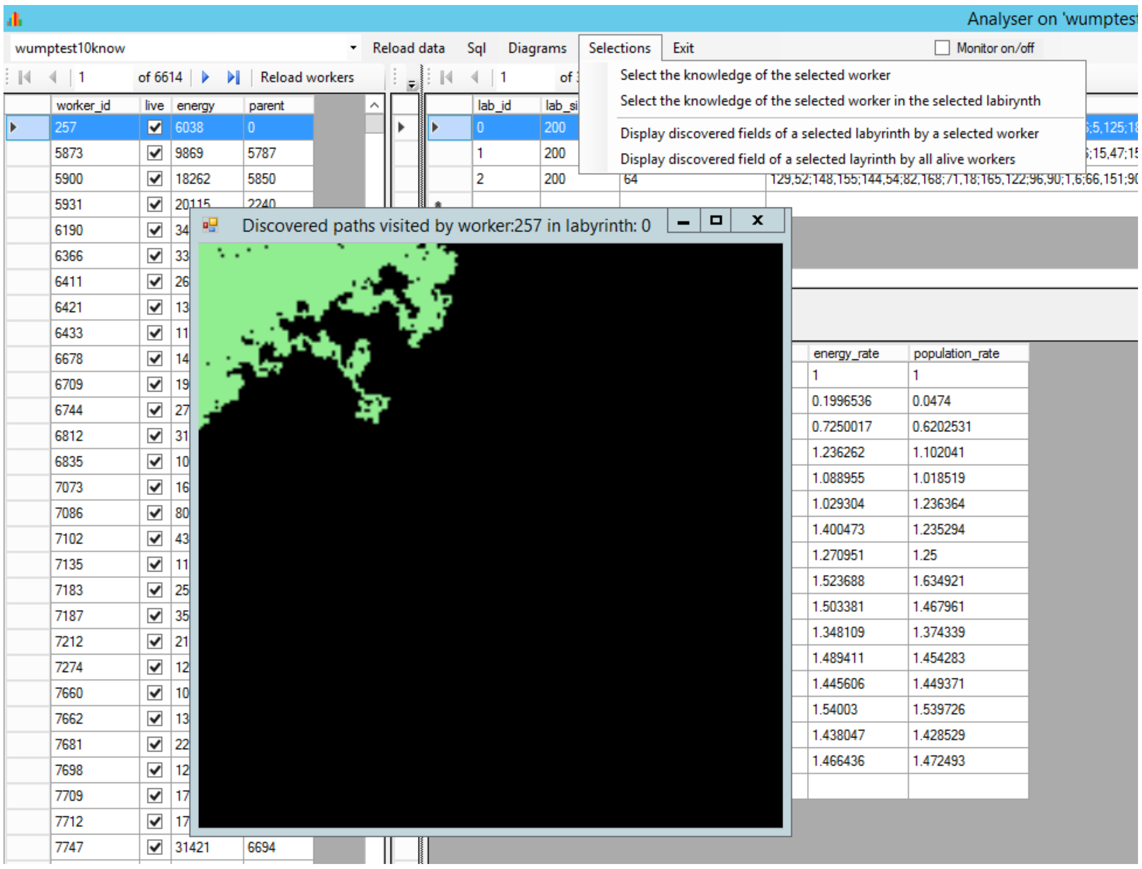

Analyser on
Selections
Exit
Monitor on/off
Select the knowledge of the selected worker
Select the knowledge of the selected worker in the selected labyrinth
Display discovered fields of a selected labyrinth by a selected worker
Display discovered fields of a selected labyrinth by all alive workers
Display discovered fields of a given generation

Figure 16: Selection menu: here you can select worker specific knowledge and display discovered fields.

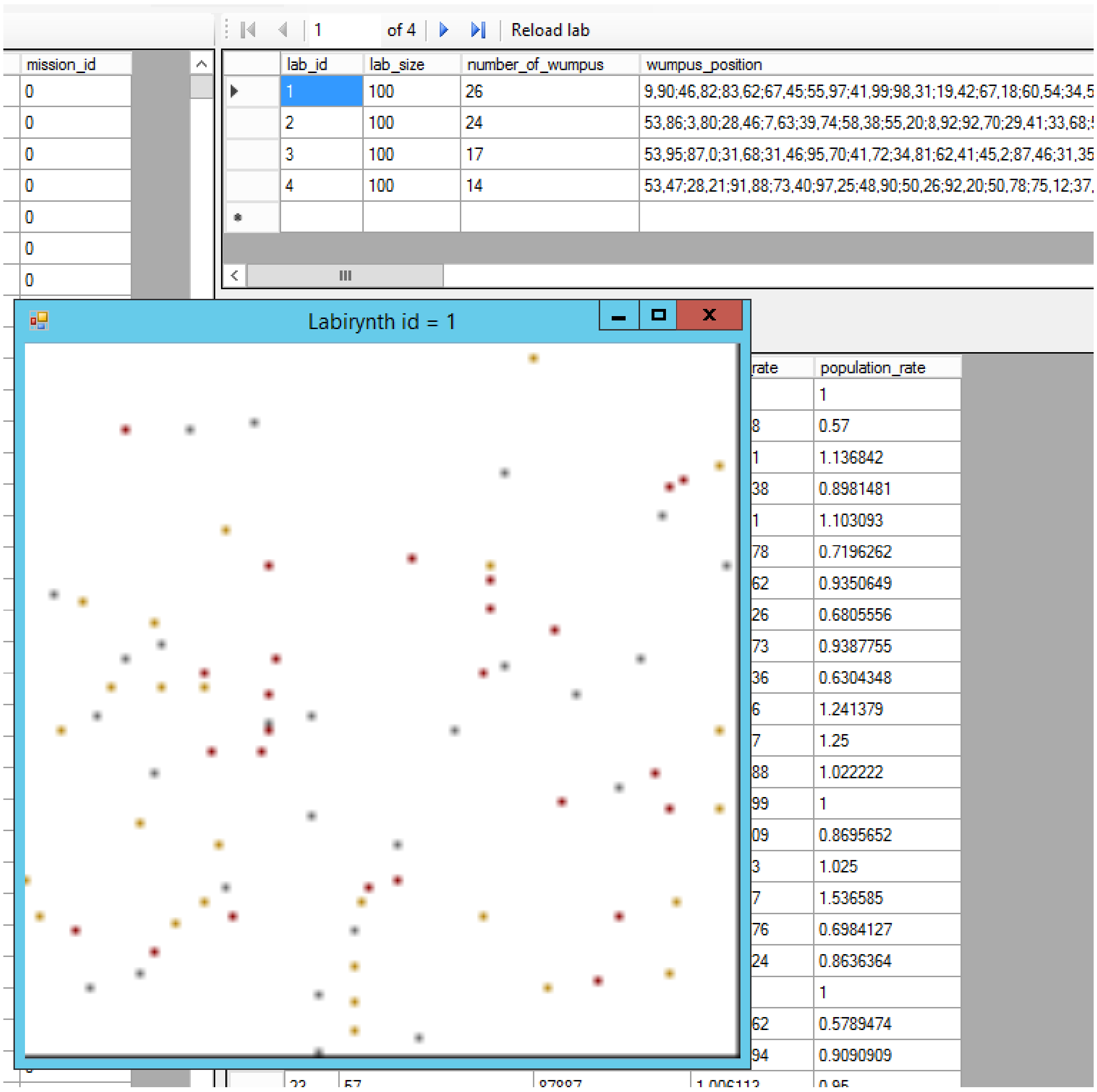

Analyser on 'wumptes
wumptest10know
Reload data
Sql
Diagrams
Selections
Exit
Monitor on/off
1
of 6614
Reload workers
1
of
Select the knowledge of the selected worker
Select the knowledge of the selected worker in the selected labirynth
Display discovered fields of a selected labyrinth by a selected worker
Display discovered field of a selected layrinth by all alive workers
Discovered paths visited by worker:257 in labyrinth: 0

worker_id	live	energy	parent
257	✓	6038	0
5873	✓	9869	5787
5900	✓	18262	5850
5931	✓	20115	2240
6190	✓	34	
6366	✓	33	
6411	✓	26	
6421	✓	13	
6433	✓	11	
6678	✓	14	
6709	✓	19	
6744	✓	27	
6812	✓	31	
6835	✓	10	
7073	✓	16	
7086	✓	80	
7102	✓	43	
7135	✓	11	
7183	✓	25	
7187	✓	35	
7212	✓	21	
7274	✓	12	
7660	✓	10	
7662	✓	13	
7681	✓	22	
7698	✓	12	
7709	✓	17	
7712	✓	17	
7747	✓	31421	6694

lab_id	lab_si		
0	200		;5,125;18
1	200		;15,47;15
2	200	64	129,52;148,155;144,54;82,168;71,18;165,122;96,90;1,6;66,151;90

energy_rate	population_rate
1	1
0.1996536	0.0474
0.7250017	0.6202531
1.236262	1.102041
1.088955	1.018519
1.029304	1.236364
1.400473	1.235294
1.270951	1.25
1.523688	1.634921
1.503381	1.467961
1.348109	1.374339
1.489411	1.454283
1.445606	1.449371
1.54003	1.539726
1.438047	1.428529
1.466436	1.472493

Figure 17: Discovered fields of a selected labyrinth

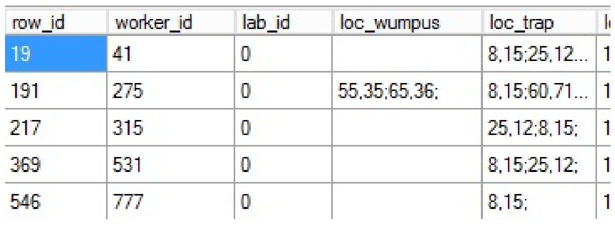


mission_id
0
0
0
0
0
0
0

lab_id	lab_size	number_of_wumpus	wumpus_position
1	100	26	9,90;46,82;83,62;67,45;55,97;41,99;98,31;19,42;67,18;60,54;34,5
2	100	24	53,86;3,80;28,46;7,63;39,74;58,38;55,20;8,92;92,70;29,41;33,68;
3	100	17	53,95;87,0;31,68;31,46;95,70;41,72;34,81;62,41;45,2;87,46;31,35
4	100	14	53,47;28,21;91,88;73,40;97,25;48,90;50,26;92,20;50,78;75,12;37,

rate	population_rate
	1
8	0.57
1	1.136842
38	0.8981481
1	1.103093
78	0.7196262
62	0.9350649
26	0.6805556
73	0.9387755
36	0.6304348
6	1.241379
7	1.25
88	1.022222
99	1
09	0.8695652
3	1.025
7	1.536585
76	0.6984127
24	0.8636364
	1
62	0.5789474
94	0.9090909

Figure 18: The selected labyrinth with many wumpuses (black points), traps (red points) and gold bars (goldenrod points). Select a labyrinth by clicking with right mouse button if you wish to see the labyrinth graphically.



row_id	worker_id	lab_id	loc_wumpus	loc_trap	l
19	41	0		8,15;25,12...	1
191	275	0	55,35;65,36;	8,15;60,71...	1
217	315	0		25,12;8,15;	1
369	531	0		8,15;25,12;	1
546	777	0		8,15;	1

Figure 19: The structure of the 'knowledge' table

7 Database tables and structures

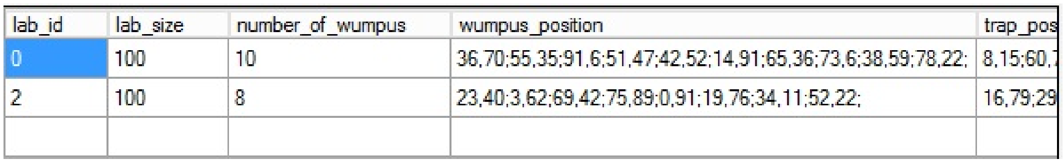
The database management system of the DEM system is Postgres. As many invasions is created as many databases is stored in Postgres. The name of every database should start with 'dem'. Every database has the following tables: knowledge, lab (labyrinths), logbook, missions, parameters, worker_path, workers.

7.1 knowledge table

- row_id: type is long integer (primary key) for uniquely identify rows
- worker_id: type is long integer (primary key) which identifies a certain worker
- lab_id: type is long integer which identifies a labyrinth
- loc_wumpus: type is string which contains the locations of wumpuses in the lab identified by lab_id separated by ','
- loc_traps: type is string which contains the locations of traps in the lab identified by lab_id separated by ','
- loc_gold: type is string which contains the locations of gold bars in the lab identified by lab_id separated by ',' (Fig. 19)

7.2 lab table

- lab_id: type is integer (primary key) for uniquely identify labyrinths
- lab_size: type is integer, which describes the rectangular (lab_size $\times$ lab_size) labyrinths



lab_id	lab_size	number_of_wumpus	wumpus_position	trap_pos
0	100	10	36,70;55,35;91,6;51,47;42,52;14,91;65,36;73,6;38,59;78,22;	8,15;60,
2	100	8	23,40;3,62;69,42;75,89;0,91;19,76;34,11;52,22;	16,79;29

Figure 20: The structure of the 'lab' table

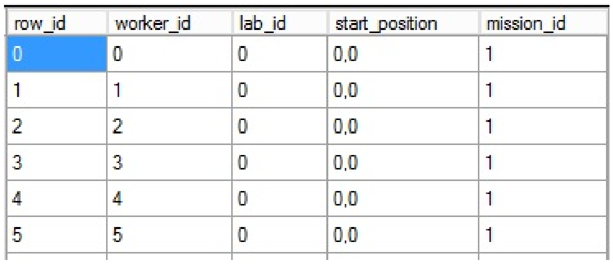
- number_of_wumpuses: type is integer which gives the number of wumpuses in the certain labyrinth
- wumpus_position: type is string which contains the positions of wumpuses separated by ','
- traps_position: type is string which contains the positions of traps separated by ','
- gold_position: type is string which contains the positions of gold bars separated by ',' (Fig. 20)

7.3 logbook table

- row_id: type is long integer (primary key) for uniquely identify rows
- worker_id: type is long integer for worker identification
- lab_id: type is integer, for labyrinth identification
- start_position: type is string which stores the start position of this mission
- mission_id: type is integer where the current mission_id is stored (Fig. 21)

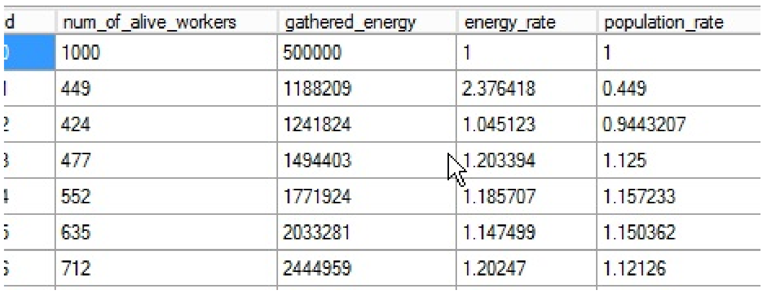
7.4 mission table

- id: type is integer (primary key) for uniquely identify missions
- number_of_alive_workers: type is long integer. This field contains the number of alive workers after the given mission



row_id	worker_id	lab_id	start_position	mission_id
0	0	0	0,0	1
1	1	0	0,0	1
2	2	0	0,0	1
3	3	0	0,0	1
4	4	0	0,0	1
5	5	0	0,0	1

Figure 21: The structure of the 'logbook' table



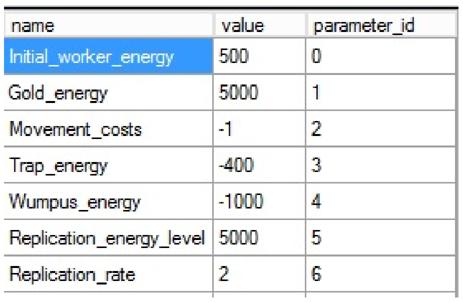
d	num_of_alive_workers	gathered_energy	energy_rate	population_rate
0	1000	500000	1	1
1	449	1188209	2.376418	0.449
2	424	1241824	1.045123	0.9443207
3	477	1494403	1.203394	1.125
4	552	1771924	1.185707	1.157233
5	635	2033281	1.147499	1.150362
6	712	2444959	1.20247	1.12126

Figure 22: The structure of the 'mission' table

- gathered_energy: type is long integer. This field contains the gathered energy by the alive workers after the given mission
- energy_rate: type is float. It's value is (the sum of workers' energy at the beginning a given mission)/(the sum of workers' energy at the end of mission)
- population_rate: type is float. It's value is (the number of workers' at the beginning of a given mission)/(the number of workers' at the end of a given mission) (Fig. 22)

7.5 parameters table

This table is a special table which stores those parameters which adjust behavior of the DCMaster program such as initial worker energy, gold energy content, movement costs, traps energy content (it is negative), wumpus energy content (it is negative), replication energy level (below this value the replication is not allowed), replication rate (in case of value 2 a worker produces 2 predecessors, in case of value 3 a worker produce 3 predecessors)



name	value	parameter_id
Initial_worker_energy	500	0
Gold_energy	5000	1
Movement_costs	-1	2
Trap_energy	-400	3
Wumpus_energy	-1000	4
Replication_energy_level	5000	5
Replication_rate	2	6

Figure 23: The structure of the 'parameters' table

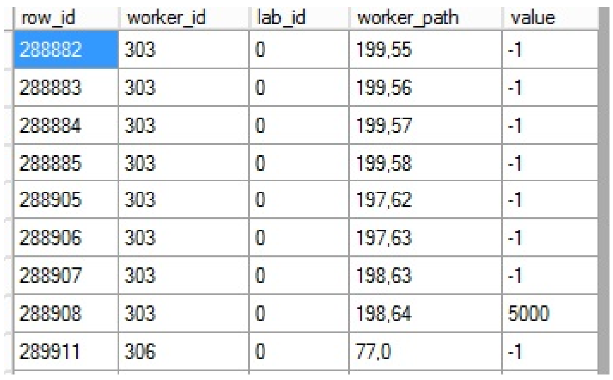
- name: type is string. The contents is name of the parameter
- value: type is integer, which contains the certain value of this parameter
- parameter_id: type is integer (primary key) for identifying parameters (Fig. 23)

7.6 worker_path table

- row_id: type is long integer (primary key) for uniquely identify the rows
- worker_id: type is long integer which identifies the workers
- lab_id: type is integer which identifies the labs
- worker_path: type is string which stores the visited position of a worker in a certain labyrinth
- value: type is integer which contains the energy content of the visited field (Fig. 24)

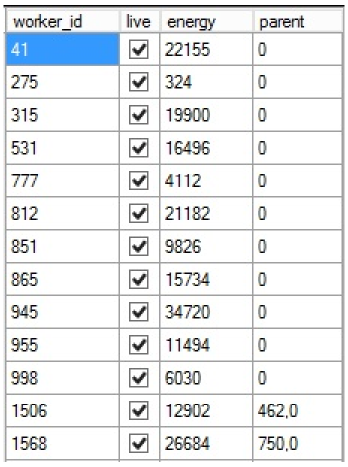
7.7 workers table

- worker_id: type is long integer (primary key) for uniquely identify workers
- live: type is boolean, which is the status (live or dead) of a certain worker
- energy: type is long integer which stores the energy content of a certain worker



row_id	worker_id	lab_id	worker_path	value
288882	303	0	199,55	-1
288883	303	0	199,56	-1
288884	303	0	199,57	-1
288885	303	0	199,58	-1
288905	303	0	197,62	-1
288906	303	0	197,63	-1
288907	303	0	198,63	-1
288908	303	0	198,64	5000
289911	306	0	77,0	-1

Figure 24: The structure of the 'worker_path' table



worker_id	live	energy	parent
41	✓	22155	0
275	✓	324	0
315	✓	19900	0
531	✓	16496	0
777	✓	4112	0
812	✓	21182	0
851	✓	9826	0
865	✓	15734	0
945	✓	34720	0
955	✓	11494	0
998	✓	6030	0
1506	✓	12902	462,0
1568	✓	26684	750,0

Figure 25: The structure of the 'workers' table

- parent: type is string which contains the ancestors' worker_ids separated by ',' (Fig. 25)