

Applying Computational Science Project 2

Generated by Doxygen 1.8.17

1 Class Index	1
1.1 Class List	1
2 File Index	3
2.1 File List	3
3 Class Documentation	5
3.1 CCircuit Class Reference	5
3.1.1 Detailed Description	6
3.1.2 Constructor & Destructor Documentation	6
3.1.2.1 CCircuit() [1/2]	6
3.1.2.2 CCircuit() [2/2]	7
3.1.3 Member Function Documentation	7
3.1.3.1 Check_Velocity()	7
3.1.3.2 Evaluate_Circuit()	7
3.1.3.3 mark_units()	8
3.1.3.4 vector2units() [1/2]	8
3.1.3.5 vector2units() [2/2]	8
3.1.4 Member Data Documentation	8
3.1.4.1 conc_toend	9
3.1.4.2 initial_conc	9
3.1.4.3 initial_tails	9
3.1.4.4 max_iterations	9
3.1.4.5 start	9
3.1.4.6 tails_toend	9
3.1.4.7 tolerance	9
3.1.4.8 units	10
3.2 CUnit Class Reference	10
3.2.1 Detailed Description	10
3.2.2 Member Function Documentation	10
3.2.2.1 set_values()	11
3.2.3 Member Data Documentation	11
3.2.3.1 conc_conc	11
3.2.3.2 conc_num	11
3.2.3.3 conc_tails	11
3.2.3.4 flow_conc	11
3.2.3.5 flow_conc_old	11
3.2.3.6 flow_tails	11
3.2.3.7 flow_tails_old	12
3.2.3.8 mark	12
3.2.3.9 tails_conc	12
3.2.3.10 tails_num	12
3.2.3.11 tails_tails	12

4 File Documentation	13
4.1 includes/CCircuit.h File Reference	13
4.1.1 Detailed Description	14
4.1.2 Variable Documentation	14
4.1.2.1 num_units	14
4.2 includes/CUnit.h File Reference	15
4.2.1 Detailed Description	16
4.2.2 Variable Documentation	16
4.2.2.1 K_conc	16
4.2.2.2 K_tails	16
4.2.2.3 phi	16
4.2.2.4 rho	16
4.2.2.5 V	17
4.3 includes/Genetic_Algorithm.h File Reference	17
4.3.1 Detailed Description	18
4.3.2 Function Documentation	18
4.3.2.1 Genetic_Algorithm()	18
4.4 src/CCircuit.cpp File Reference	19
4.4.1 Detailed Description	19
4.5 src/CUnit.cpp File Reference	20
4.5.1 Detailed Description	20
4.6 src/Genetic_Algorithm.cpp File Reference	21
4.6.1 Detailed Description	22
4.6.2 Function Documentation	22
4.6.2.1 best_parent2child()	22
4.6.2.2 calculate_fitness_value()	22
4.6.2.3 create_chromosome_set()	24
4.6.2.4 create_parent()	24
4.6.2.5 crossover()	24
4.6.2.6 Genetic_Algorithm()	25
4.6.2.7 get_rand()	25
4.6.2.8 mutate()	25
4.6.2.9 select_parent()	26
4.7 src/main.cpp File Reference	26
4.7.1 Detailed Description	27
4.7.2 Function Documentation	27
4.7.2.1 main()	27
Index	29

Chapter 1

Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

CCircuit	Circuit made up of units connected to each other, constructed from chromosome array or vector	5
CUnit	Units that make up a circuit	10

Chapter 2

File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

includes/ CCircuit.h	
Header file for the CCircuit class	13
includes/ CUnit.h	
Header file for the CUnit class	15
includes/ Genetic_Algorithm.h	
For genetic algorithm function declearation	17
src/ CCircuit.cpp	
Encapsulate test functions	19
src/ CUnit.cpp	
Calculation of products (Gormanium) and wastes	20
src/ Genetic_Algorithm.cpp	
For genetic algorithm implementation	21
src/ main.cpp	
Main file for running main function	26

Chapter 3

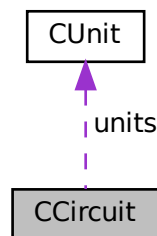
Class Documentation

3.1 CCircuit Class Reference

Circuit made up of units connected to each other, constructed from chromosome array or vector.

```
#include <CCircuit.h>
```

Collaboration diagram for CCircuit:



Public Member Functions

- `CCircuit` (`std::vector< int >` chromosome, double `tolerance=1e-6`, int `max_iterations=1000`, int `initial_conc=10`, int `initial_tails=100`)
Constructor for `CCircuit` object from circuit vector.
- `CCircuit` (int *chromosome, double `tolerance=1e-6`, int `max_iterations=1000`, int `initial_conc=10`, int `initial_tails=100`)
Constructor for `CCircuit` object from circuit vector stored as an integer array.
- bool `Check_Validity` ()
check validity of circuit
- double `Evaluate_Circuit` (double `tolerance=1e-6`, int `max_iterations=1000`)
Score a circuit based on its performance.

Private Member Functions

- void [vector2units](#) (std::vector< int > chromosome)
Transfer data from circuit vector to a vector of units i.e. circuit.
- void [vector2units](#) (int *chromosome)
Transfer data from circuit vector stored as an integer array to a vector of units i.e. circuit.
- void [mark_units](#) (int unit_num)
Traverse circuit and mark units that have been passed.

Private Attributes

- [CUnit](#) units [num_units]
- int [start](#)
- bool [conc_toend](#)
- bool [tails_toend](#)
- double [tolerance](#)
- int [max_iterations](#)
- double [initial_conc](#)
- double [initial_tails](#)

3.1.1 Detailed Description

Circuit made up of units connected to each other, constructed from chromosome array or vector.

3.1.2 Constructor & Destructor Documentation

3.1.2.1 CCircuit() [1/2]

```
CCircuit::CCircuit (
    std::vector< int > chromosome,
    double tolerance = 1e-6,
    int max_iterations = 1000,
    int initial_conc = 10,
    int initial_tails = 100 )
```

Constructor for [CCircuit](#) object from circuit vector.

Parameters

<i>chromosome</i>	circuit vector
<i>tolerance</i>	error tolerance
<i>max_iterations</i>	maximum number of iterations
<i>initial_conc</i>	initial feed concentrate
<i>initial_tails</i>	initial feed tailings

3.1.2.2 CCircuit() [2/2]

```
CCircuit::CCircuit (
    int * chromosome,
    double tolerance = 1e-6,
    int max_iterations = 1000,
    int initial_conc = 10,
    int initial_tails = 100 )
```

Constructor for [CCircuit](#) object from circuit vector stored as an integer array.

Parameters

<i>chromosome</i>	circuit vector stored as an integer array
<i>tolerance</i>	error tolerance
<i>max_iterations</i>	maximum number of iterations
<i>initial_conc</i>	initial feed concentrate
<i>initial_tails</i>	initial feed tailings

3.1.3 Member Function Documentation

3.1.3.1 Check_Validity()

```
bool CCircuit::Check_Validity ( )
```

check validity of circuit

Returns

bool true if the vector is valid, false if the vector is invalid

3.1.3.2 Evaluate_Circuit()

```
double CCircuit::Evaluate_Circuit (
    double tolerance = 1e-6,
    int max_iterations = 1000 )
```

Score a circuit based on its performance.

Parameters

<i>tolerance</i>	Tolerance
<i>max_iterations</i>	Maximum number of iterations

Returns

double Score

3.1.3.3 mark_units()

```
void CCircuit::mark_units (
    int unit_num ) [private]
```

Traverse circuit and mark units that have been passed.

Parameters

<i>unit_num</i>	unit number that is marked
-----------------	----------------------------

3.1.3.4 vector2units() [1/2]

```
void CCircuit::vector2units (
    int * chromosome ) [private]
```

Transfer data from circuit vector stored as an integer array to a vector of units i.e. circuit.

Parameters

<i>chromosome</i>	Circuit vector stored as an integer array
-------------------	---

3.1.3.5 vector2units() [2/2]

```
void CCircuit::vector2units (
    std::vector< int > chromosome ) [private]
```

Transfer data from circuit vector to a vector of units i.e. circuit.

Parameters

<i>chromosome</i>	Circuit vector
-------------------	----------------

3.1.4 Member Data Documentation

3.1.4.1 conc_toend

```
bool CCircuit::conc_toend [private]
```

True if concentrate outlet is found

3.1.4.2 initial_conc

```
double CCircuit::initial_conc [private]
```

Initial value of concentrate outlet

3.1.4.3 initial_tails

```
double CCircuit::initial_tails [private]
```

Initial value of tailings outlet

3.1.4.4 max_iterations

```
int CCircuit::max_iterations [private]
```

Maximum number of iterations

3.1.4.5 start

```
int CCircuit::start [private]
```

Feed unit number

3.1.4.6 tails_toend

```
bool CCircuit::tails_toend [private]
```

True if tailings outlet is found

3.1.4.7 tolerance

```
double CCircuit::tolerance [private]
```

tolerance for error in concentrate flow an tailings flow

3.1.4.8 units

```
CUnit CCircuit::units[num_units] [private]
```

Array of units of length num_units. Build up the circuit.

The documentation for this class was generated from the following files:

- includes/CCircuit.h
- src/CCircuit.cpp

3.2 CUnit Class Reference

Units that make up a circuit.

```
#include <CUnit.h>
```

Public Member Functions

- void [set_values](#) ()
Calculate physical parameters for each iteration.

Public Attributes

- int [conc_num](#)
- int [tails_num](#)
- bool [mark](#) = false
- double [flow_conc](#) = 0
- double [flow_tails](#) = 0
- double [flow_conc_old](#) = 0
- double [flow_tails_old](#) = 0
- double [conc_conc](#)
- double [conc_tails](#)
- double [tails_conc](#)
- double [tails_tails](#)

3.2.1 Detailed Description

Units that make up a circuit.

3.2.2 Member Function Documentation

3.2.2.1 set_values()

```
void CUnit::set_values ( )
```

Calculate physical parameters for each iteration.

3.2.3 Member Data Documentation

3.2.3.1 conc_conc

```
double CUnit::conc_conc
```

concentrate in concentrate stream

3.2.3.2 conc_num

```
int CUnit::conc_num
```

index of the unit to which this unit's concentrate stream is connected

3.2.3.3 conc_tails

```
double CUnit::conc_tails
```

tailings in concentrate stream

3.2.3.4 flow_conc

```
double CUnit::flow_conc = 0
```

the mass flow rate of solid (gormanium)

3.2.3.5 flow_conc_old

```
double CUnit::flow_conc_old = 0
```

the mass flow rate of solid (gormanium) from previous iteration

3.2.3.6 flow_tails

```
double CUnit::flow_tails = 0
```

the mass flow rate of solid (waste)

3.2.3.7 flow_tails_old

```
double CUnit::flow_tails_old = 0
```

the mass flow rate of solid (waste) from previous iteration

3.2.3.8 mark

```
bool CUnit::mark = false
```

A Boolean that is changed to true if the unit has been seen

3.2.3.9 tails_conc

```
double CUnit::tails_conc
```

concentrate in tailings stream

3.2.3.10 tails_num

```
int CUnit::tails_num
```

index of the unit to which this unit's tailings stream is connected

3.2.3.11 tails_tails

```
double CUnit::tails_tails
```

tailings in tailings stream

The documentation for this class was generated from the following files:

- includes/[CUnit.h](#)
- src/[CUnit.cpp](#)

Chapter 4

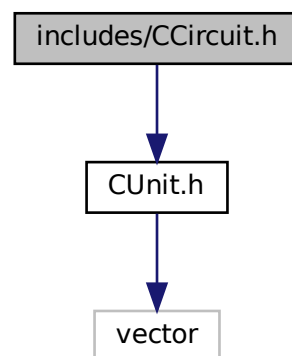
File Documentation

4.1 includes/CCircuit.h File Reference

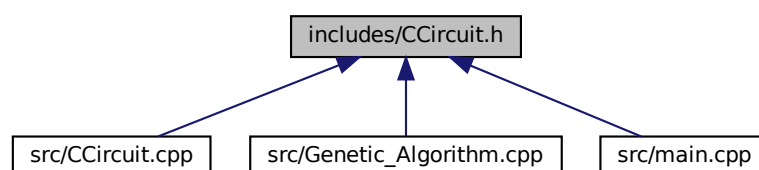
Header file for the [CCircuit](#) class.

```
#include "CUnit.h"
```

Include dependency graph for CCircuit.h:



This graph shows which files directly or indirectly include this file:



Classes

- class [CCircuit](#)

Circuit made up of units connected to each other, constructed from chromosome array or vector.

Variables

- const int [num_units](#) = 10

4.1.1 Detailed Description

Header file for the [CCircuit](#) class.

Author

Xiao Teng

Version

0.2

Date

2022-03-25

Copyright

Copyright (c) 2022

4.1.2 Variable Documentation

4.1.2.1 num_units

```
const int num_units = 10
```

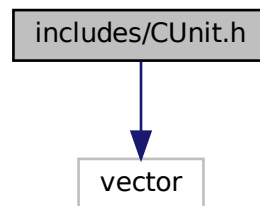
Number of units. This is constant and default value is 10

4.2 includes/CUnit.h File Reference

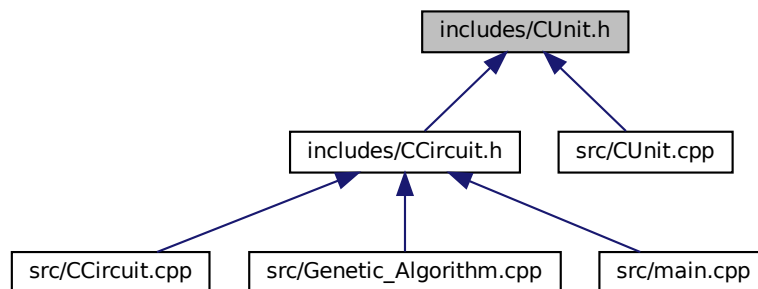
Header file for the [CUnit](#) class.

```
#include <vector>
```

Include dependency graph for CUnit.h:



This graph shows which files directly or indirectly include this file:



Classes

- class [CUnit](#)
Units that make up a circuit.

Variables

- const double [K_tails](#) = 0.0005
- const double [K_conc](#) = 0.005
- const double [rho](#) = 3000
- const double [phi](#) = 0.1
- const double [V](#) = 10

4.2.1 Detailed Description

Header file for the [CUnit](#) class.

Author

Xiao Teng

Version

0.2

Date

2022-03-25

Copyright

Copyright (c) 2022

4.2.2 Variable Documentation

4.2.2.1 K_conc

```
const double K_conc = 0.005
```

the rate constant of concentrate (gormanium)

4.2.2.2 K_tails

```
const double K_tails = 0.0005
```

the rate constant tailings (waste)

4.2.2.3 phi

```
const double phi = 0.1
```

the total solids (gormanium + waste) content of the feed by volume

4.2.2.4 rho

```
const double rho = 3000
```

solids (gormanium + waste) density

4.2.2.5 V

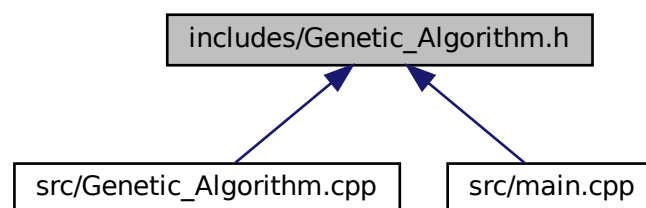
```
const double V = 10
```

volume of each cell/unit

4.3 includes/Genetic_Algorithm.h File Reference

For genetic algorithm function declaration.

This graph shows which files directly or indirectly include this file:



Macros

- `#define NUM_PARENT 150`
- `#define NUM_UNIT 10`
- `#define TOLERANCE 0.001`
- `#define MAX_ITERATIONS 500`
- `#define NUM_CHILDREN 100`
- `#define CROSSOVER_PRO 0.95`
- `#define MUTATE_PRO 0.01`
- `#define MAX_EVOLUTIONS 3000`

Functions

- double `Genetic_Algorithm` (void)
Produce child vectors from a list of parent vectors.

4.3.1 Detailed Description

For genetic algorithm function declearation.

Author

Galena Group, Yang Bai, Tengteng Huang, Xiao Teng

Version

0.5

Date

2022-03-25

Copyright

Copyright (c) 2022

4.3.2 Function Documentation

4.3.2.1 Genetic_Algorithm()

```
double Genetic_Algorithm (  
    void )
```

Produce child vectors from a list of parent vectors.

Returns

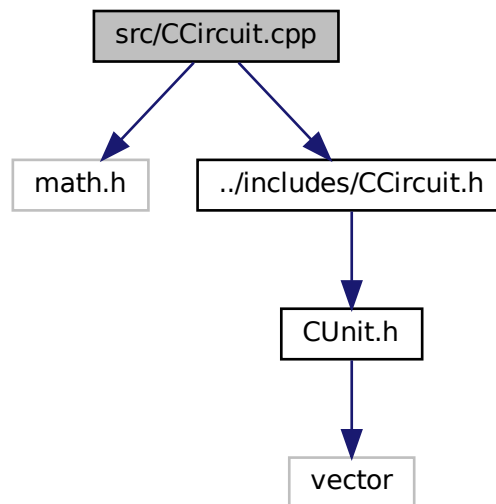
double highest score

4.4 src/CCircuit.cpp File Reference

Encapsulate test functions.

```
#include <math.h>
#include "../includes/CCircuit.h"
```

Include dependency graph for CCircuit.cpp:



4.4.1 Detailed Description

Encapsulate test functions.

Author

Xiao Teng, Ian Wang, Yuna Nakamura, Beini Zhang, Jingyu Zhou

Version

0.2

Date

2022-03-25

Copyright

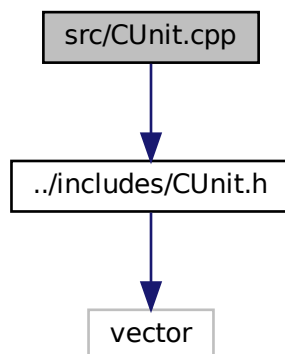
Copyright (c) 2022

4.5 src/CUnit.cpp File Reference

Calculation of products (Gormanium) and wastes.

```
#include "../includes/CUnit.h"
```

Include dependency graph for CUnit.cpp:



4.5.1 Detailed Description

Calculation of products (Gormanium) and wastes.

Author

Xiao Teng

Version

0.2

Date

2022-03-25

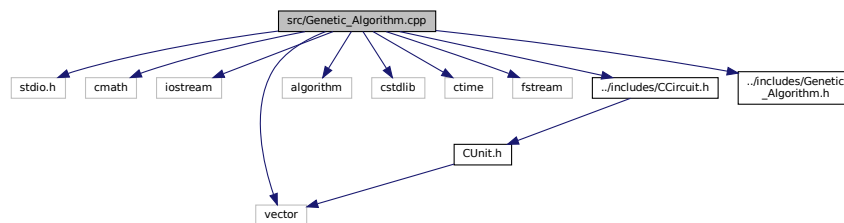
Copyright

Copyright (c) 2022

4.6 src/Genetic_Algorithm.cpp File Reference

For genetic algorithm implementation.

```
#include <stdio.h>
#include <cmath>
#include <iostream>
#include <vector>
#include <algorithm>
#include <cstdlib>
#include <ctime>
#include <fstream>
#include "../includes/CCircuit.h"
#include "../includes/Genetic_Algorithm.h"
Include dependency graph for Genetic_Algorithm.cpp:
```



Functions

- void [create_parent](#) (vector< int > *parent, int unit_num)
Fill in parent vector by randomly generating numbers.
- void [create_chromosome_set](#) (vector< vector< int > > *parent_set, int unit_num, int set_num)
Create a set of parent vectors.
- void [calculate_fitness_value](#) (vector< double > *score, vector< vector< int > > parent_set, double tolerance, int max_iterations)
Calculate fitness value for parent set.
- double [best_parent2child](#) (vector< vector< int > > &child_set, vector< double > score, vector< vector< int > > parent_set)
Select best parent in parent set and put it into child set.
- int [select_parent](#) (vector< vector< int > > parent_set, vector< double > score)
Randomly select a parent.
- static int [get_rand](#) (double x)
Judge whether the event occurs by given probability.
- void [crossover](#) (vector< int > &father, vector< int > &mother)
Crossover: Swap a portion of one parent vector with a portion of another parent vector.
- void [mutate](#) (vector< int > &before)
Mutate: Random changes in the numbers in the vector.
- double [Genetic_Algorithm](#) (void)
Produce child vectors from a list of parent vectors.

4.6.1 Detailed Description

For genetic algorithm implementation.

Author

Galena Group, Yang Bai, Tengteng Huang, Xiao Teng

Version

0.5

Date

2022-03-25

Copyright

Copyright (c) 2022

4.6.2 Function Documentation

4.6.2.1 best_parent2child()

```
double best_parent2child (
    vector< vector< int > > & child_set,
    vector< double > score,
    vector< vector< int > > parent_set )
```

Select best parent in parent set and put it into child set.

Parameters

<i>child_set</i>	Vector for loading child vector
<i>score</i>	Vector for fitness value
<i>parent_set</i>	Vector for Parents set

4.6.2.2 calculate_fitness_value()

```
void calculate_fitness_value (
    vector< double > * score,
    vector< vector< int > > parent_set,
```

```
double tolerance,  
int max_iterations )
```

Calculate fitness value for parent set.

Parameters

<i>score</i>	Empty vector for loading fitness value
<i>parent_set</i>	Parent vector that already load
<i>tolerance</i>	Error tolerance
<i>max_iterations</i>	Maximum number of iterations

4.6.2.3 create_chromosome_set()

```
void create_chromosome_set (
    vector< vector< int > > * parent_set,
    int unit_num,
    int set_num )
```

Create a set of parent vectors.

Parameters

<i>parent_set</i>	Empty vector for loading multiple parents
<i>unit_num</i>	Number of circuit units
<i>set_num</i>	Number of Parents

4.6.2.4 create_parent()

```
void create_parent (
    vector< int > * parent,
    int unit_num )
```

Fill in parent vector by randomly generating numbers.

Parameters

<i>parent</i>	Empty vector for loading parent
<i>unit_num</i>	Number of circuit units

4.6.2.5 crossover()

```
void crossover (
    vector< int > & father,
    vector< int > & mother )
```

Crossover: Swap a portion of one parent vector with a portion of another parent vector.

Parameters

<i>father</i>	One parent vector
<i>mother</i>	Another parent vector

4.6.2.6 Genetic_Algorithm()

```
double Genetic_Algorithm (
    void )
```

Produce child vectors from a list of parent vectors.

Returns

double highest score

4.6.2.7 get_rand()

```
static int get_rand (
    double x ) [static]
```

Judge whether the event occurs by given probability.

Parameters

<i>x</i>	Probability
----------	-------------

Returns

int whether the event occurs by given probability

4.6.2.8 mutate()

```
void mutate (
    vector< int > & before )
```

Mutate: Random changes in the numbers in the vector.

Parameters

<i>before</i>	Vector to mutate
---------------	------------------

4.6.2.9 select_parent()

```
int select_parent (
    vector< vector< int > > parent_set,
    vector< double > score )
```

Randomly select a parent.

Parameters

<i>parent_set</i>	Vector for Parent set
<i>score</i>	Vector for fitness value

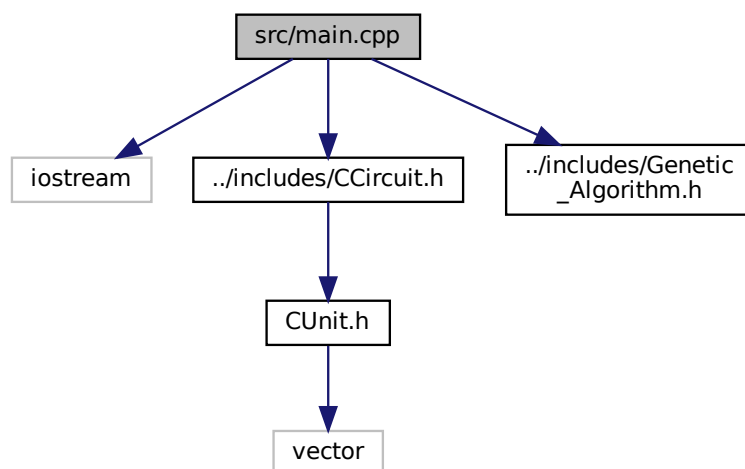
Returns

int Parent number in the parents set

4.7 src/main.cpp File Reference

main file for running main function

```
#include <iostream>
#include "../includes/CCircuit.h"
#include "../includes/Genetic_Algorithm.h"
Include dependency graph for main.cpp:
```



Functions

- `int main ()`

Example of how to use Genertic Algorithm, If you want to change parameters, go to [Genetic_Algorithm.h](#).

4.7.1 Detailed Description

main file for running main function

Author

Yang Bai

Version

0.1

Date

2022-03-25

Copyright

Copyright (c) 2022

4.7.2 Function Documentation

4.7.2.1 `main()`

```
int main ( )
```

Example of how to use Genertic Algorithm, If you want to change parameters, go to [Genetic_Algorithm.h](#).

Returns

int

Index

- best_parent2child
 - Genetic_Algorithm.cpp, [22](#)
- calculate_fitness_value
 - Genetic_Algorithm.cpp, [22](#)
- CCircuit, [5](#)
 - CCircuit, [6](#)
 - Check_Velocity, [7](#)
 - conc_toend, [8](#)
 - Evaluate_Circuit, [7](#)
 - initial_conc, [9](#)
 - initial_tails, [9](#)
 - mark_units, [8](#)
 - max_iterations, [9](#)
 - start, [9](#)
 - tails_toend, [9](#)
 - tolerance, [9](#)
 - units, [9](#)
 - vector2units, [8](#)
- CCircuit.h
 - num_units, [14](#)
- Check_Velocity
 - CCircuit, [7](#)
- conc_conc
 - CUnit, [11](#)
- conc_num
 - CUnit, [11](#)
- conc_tails
 - CUnit, [11](#)
- conc_toend
 - CCircuit, [8](#)
- create_chromosome_set
 - Genetic_Algorithm.cpp, [24](#)
- create_parent
 - Genetic_Algorithm.cpp, [24](#)
- crossover
 - Genetic_Algorithm.cpp, [24](#)
- CUnit, [10](#)
 - conc_conc, [11](#)
 - conc_num, [11](#)
 - conc_tails, [11](#)
 - flow_conc, [11](#)
 - flow_conc_old, [11](#)
 - flow_tails, [11](#)
 - flow_tails_old, [11](#)
 - mark, [12](#)
 - set_values, [10](#)
 - tails_conc, [12](#)
 - tails_num, [12](#)
 - tails_tails, [12](#)
- CUnit.h
 - K_conc, [16](#)
 - K_tails, [16](#)
 - phi, [16](#)
 - rho, [16](#)
 - V, [16](#)
- Evaluate_Circuit
 - CCircuit, [7](#)
- flow_conc
 - CUnit, [11](#)
- flow_conc_old
 - CUnit, [11](#)
- flow_tails
 - CUnit, [11](#)
- flow_tails_old
 - CUnit, [11](#)
- Genetic_Algorithm
 - Genetic_Algorithm.cpp, [25](#)
 - Genetic_Algorithm.h, [18](#)
- Genetic_Algorithm.cpp
 - best_parent2child, [22](#)
 - calculate_fitness_value, [22](#)
 - create_chromosome_set, [24](#)
 - create_parent, [24](#)
 - crossover, [24](#)
 - Genetic_Algorithm, [25](#)
 - get_rand, [25](#)
 - mutate, [25](#)
 - select_parent, [26](#)
- Genetic_Algorithm.h
 - Genetic_Algorithm, [18](#)
- get_rand
 - Genetic_Algorithm.cpp, [25](#)
- includes/CCircuit.h, [13](#)
- includes/CUnit.h, [15](#)
- includes/Genetic_Algorithm.h, [17](#)
- initial_conc
 - CCircuit, [9](#)
- initial_tails
 - CCircuit, [9](#)
- K_conc
 - CUnit.h, [16](#)
- K_tails
 - CUnit.h, [16](#)
- main

- main.cpp, [27](#)
- main.cpp
 - main, [27](#)
- mark
 - CUnit, [12](#)
- mark_units
 - CCircuit, [8](#)
- max_iterations
 - CCircuit, [9](#)
- mutate
 - Genetic_Algorithm.cpp, [25](#)
- num_units
 - CCircuit.h, [14](#)
- phi
 - CUnit.h, [16](#)
- rho
 - CUnit.h, [16](#)
- select_parent
 - Genetic_Algorithm.cpp, [26](#)
- set_values
 - CUnit, [10](#)
- src/CCircuit.cpp, [19](#)
- src/CUnit.cpp, [20](#)
- src/Genetic_Algorithm.cpp, [21](#)
- src/main.cpp, [26](#)
- start
 - CCircuit, [9](#)
- tails_conc
 - CUnit, [12](#)
- tails_num
 - CUnit, [12](#)
- tails_tails
 - CUnit, [12](#)
- tails_toend
 - CCircuit, [9](#)
- tolerance
 - CCircuit, [9](#)
- units
 - CCircuit, [9](#)
- V
 - CUnit.h, [16](#)
- vector2units
 - CCircuit, [8](#)