Applying Computational Science Project 2

Generated by Doxygen 1.8.17

Chapter 1

Class Index

1.1 Class List

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

Chapter 2

File Index

2.1 File List

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

Chapter 3

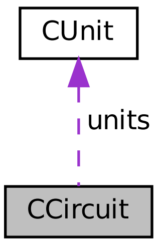
Class Documentation

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

chromosome	circuit vector
tolerance	error tolerance
max_iterations	maximum number of iterations
initial_conc	initial feed concentrate
initial_tails	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

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

tolerance	Tolerance
max_iterations	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

unit_num	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

chromosome	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

chromosome	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

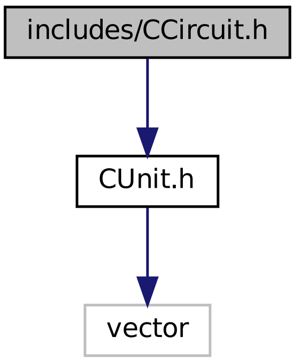
File Documentation

4.1 includes/CCircuit.h File Reference

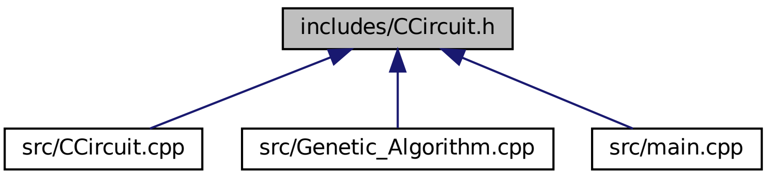
Header file for the CCircuit class.

```
#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
```

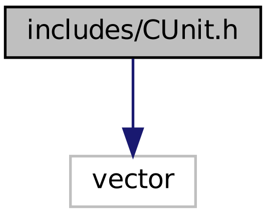
Number of units. This is constant and default value is 10

4.2 includes/CUnit.h File Reference

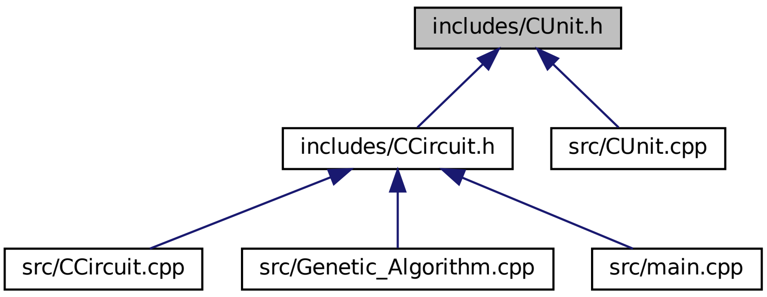
Header file for the CUnit class.

```
#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

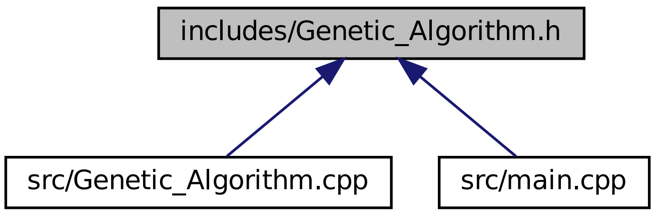
```
const double V = 10
```

volume of each cell/unit

4.3 includes/Genetic_Algorithm.h File Reference

For genetic algorithm function declartion.

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 decleartion.

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

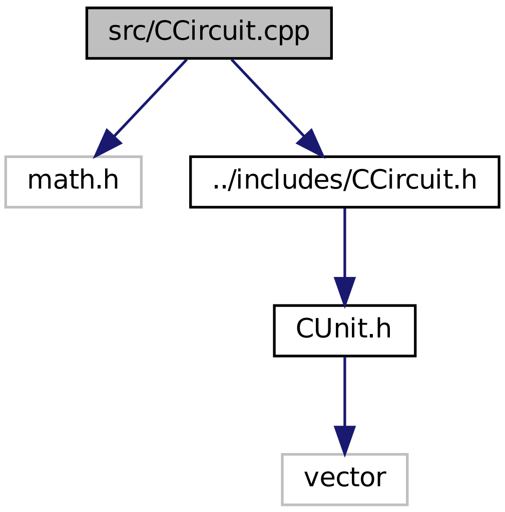
double highest score

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

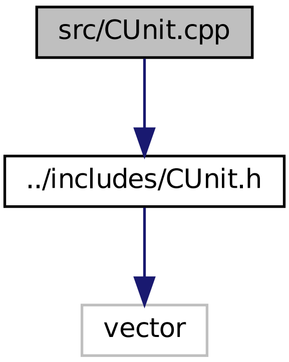
Copyright

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

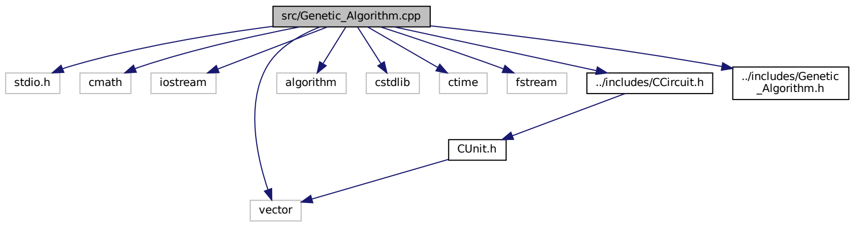
Copyright (c) 2022

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

child_set	Vector for loading child vector
score	Vector for fitness value
parent_set	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

score	Empty vector for loading fitness value
parent_set	Parent vector that already load
tolerance	Error tolerance
max_iterations	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

parent_set	Empty vector for loading multiple parents
unit_num	Number of circuit units
set_num	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

parent	Empty vector for loading parent
unit_num	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

father	One parent vector
mother	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

x	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

before	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

parent_set	Vector for Parent set
score	Vector for fitness value

Returns

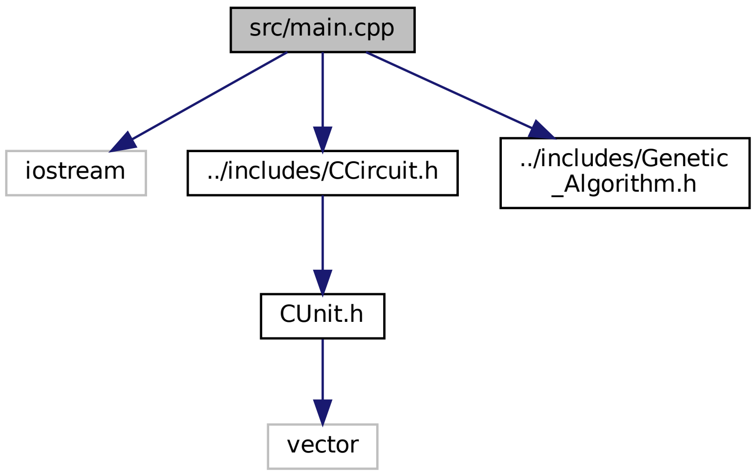
int Parent number in the parents set

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