

Closest String Problem with ACO

Generated by Doxygen 1.8.11

Contents

1	Hierarchical Index	1
1.1	Class Hierarchy	1
2	Class Index	3
2.1	Class List	3
3	File Index	5
3.1	File List	5
4	Class Documentation	7
4.1	ACS Class Reference	7
4.1.1	Detailed Description	7
4.1.2	Member Function Documentation	7
4.1.2.1	buildSolution(Solution &solution)	7
4.1.2.2	depositPheromones(Solution &ant)	8
4.1.2.3	exploitationChoice(int pos)	8
4.1.2.4	solve(Solution &best)	8
4.2	AntColony Class Reference	8
4.2.1	Detailed Description	9
4.2.2	Constructor & Destructor Documentation	9
4.2.2.1	AntColony(Instance &instance)	9
4.2.3	Member Function Documentation	10
4.2.3.1	buildSolution(Solution &solution)	10
4.2.3.2	depositPheromones(Solution &ant)	10
4.2.3.3	randomChoice(int pos)	10

4.2.3.4	<code>solve(Solution &best)</code>	10
4.3	Instance Class Reference	11
4.3.1	Detailed Description	11
4.3.2	Member Function Documentation	11
4.3.2.1	<code>getIndexChar(int ind)</code>	11
4.3.2.2	<code>getString(int ind)</code>	12
4.3.2.3	<code>greedyScore(int pos, int ch)</code>	12
4.3.2.4	<code>load(std::string fileName)</code>	12
4.3.2.5	<code>nChar()</code>	12
4.3.2.6	<code>nString()</code>	13
4.3.2.7	<code>stringLength()</code>	13
4.4	MaxMin Class Reference	13
4.4.1	Detailed Description	14
4.4.2	Constructor & Destructor Documentation	14
4.4.2.1	<code>MaxMin(Instance &instance)</code>	14
4.4.3	Member Function Documentation	14
4.4.3.1	<code>depositPheromones(Solution &ant)</code>	14
4.4.3.2	<code>solve(Solution &best)</code>	14
4.5	Utils::Parameters Class Reference	15
4.5.1	Detailed Description	15
4.6	Solution Class Reference	15
4.6.1	Detailed Description	16
4.6.2	Constructor & Destructor Documentation	16
4.6.2.1	<code>Solution(Instance &instance)</code>	16
4.6.3	Member Function Documentation	16
4.6.3.1	<code>cost()</code>	16
4.6.3.2	<code>getChar(int pos)</code>	17
4.6.3.3	<code>operator=(const Solution &solution)</code>	17
4.6.3.4	<code>setChar(int pos, int character)</code>	17
4.6.3.5	<code>setCharUpdate(int pos, int character)</code>	17
4.7	Utils Class Reference	17
4.7.1	Detailed Description	18
4.7.2	Member Function Documentation	18
4.7.2.1	<code>randomNumber()</code>	18

5 File Documentation	19
5.1 src/ACS.hpp File Reference	19
5.2 src/AntColony.hpp File Reference	19
5.2.1 Detailed Description	19
5.3 src/Instance.hpp File Reference	19
5.4 src/MaxMin.hpp File Reference	20
5.4.1 Detailed Description	20
5.5 src/Solution.hpp File Reference	20
5.5.1 Detailed Description	20
5.6 src/Utils.hpp File Reference	20
5.6.1 Detailed Description	21
Index	23

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

AntColony	8
ACS	7
MaxMin	13
Instance	11
Utils::Parameters	15
Solution	15
Utils	17

Chapter 2

Class Index

2.1 Class List

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

ACS	Class ACS for the Ant Colony System metaheuristic	7
AntColony	Base class for ant colony metaheuristics for the Closest String Problem	8
Instance	Manage an instance of the csp problem	11
MaxMin	Class MaxMin : Max Min ant colony optimization metaheuristic for the Closes String Problem .	13
Utils::Parameters	Parameters of the program	15
Solution	Class Solution representing a solution of the Closest String Problem	15
Utils	Definition of the class Utils containing the utilites functions	17

Chapter 3

File Index

3.1 File List

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

src/ ACS.hpp	19
src/ AntColony.hpp	
Definition of the virtual class AntColony	19
src/ Instance.hpp	19
src/ MaxMin.hpp	
Definition of a class MaxMin	20
src/ Solution.hpp	
Definition of a class Solution	20
src/ Utils.hpp	
Utilities for the artificial ant framework	20

Chapter 4

Class Documentation

4.1 ACS Class Reference

class [ACS](#) for the Ant Colony System metaheuristic

```
#include <ACS.hpp>
```

Inheritance diagram for ACS:

Collaboration diagram for ACS:

Public Member Functions

- **ACS** ([Instance](#) &instance)
- virtual void [solve](#) ([Solution](#) &best)
solve the CSP problem

Protected Member Functions

- virtual void [depositPheromones](#) ([Solution](#) &ant)
deposit pheromones
- virtual void [buildSolution](#) ([Solution](#) &solution)
build a new solution based on the pheromones and the heuristic information, and deposit pheromones at the same time (local rule)
- virtual int [exploitationChoice](#) (int pos)
*return the exploitation choice for this position (the max of pheromone*heuristic_info)*
- virtual void [computeProbas](#) ()
compute the probabilities for each decision of the artificial ants and the exploitation choice too

Additional Inherited Members

4.1.1 Detailed Description

class [ACS](#) for the Ant Colony System metaheuristic

4.1.2 Member Function Documentation

4.1.2.1 virtual void [ACS::buildSolution](#) ([Solution](#) & *solution*) [protected], [virtual]

build a new solution based on the pheromones and the heuristic information, and deposit pheromones at the same time (local rule)

Parameters

<i>solution</i>	the solution built
-----------------	--------------------

Reimplemented from [AntColony](#).

4.1.2.2 virtual void ACS::depositPheromones (**Solution & ant**) [protected],[virtual]

deposit pheromones

Parameters

<i>ant</i>	the ant which deposits the pheromones
------------	---------------------------------------

Reimplemented from [AntColony](#).

4.1.2.3 virtual int ACS::exploitationChoice (int *pos*) [protected],[virtual]

return the exploitation choice for this position (the max of pheromone*heuristic_info)

Parameters

<i>pos</i>	the position to make a choice in
------------	----------------------------------

Returns

the choice made (character)

4.1.2.4 virtual void ACS::solve (**Solution & best**) [virtual]

solve the CSP problem

Parameters

<i>best</i>	the best solution found
-------------	-------------------------

Reimplemented from [AntColony](#).

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

- [src/ACS.hpp](#)

4.2 AntColony Class Reference

base class for ant colony metaheuristics for the Closest String Problem

```
#include <AntColony.hpp>
```

Inheritance diagram for AntColony:

Collaboration diagram for AntColony:

Public Member Functions

- [AntColony](#) ([Instance](#) &instance)
constructor
- virtual void [solve](#) ([Solution](#) &best)
solve the CSP problem
- virtual void [displayPheromones](#) ()
display the pheromone matrix of the algorithm
- void [displayProbas](#) ()
display the probabilities matrix of the algorithm

Protected Member Functions

- virtual void [buildSolution](#) ([Solution](#) &solution)
build a new solution based on the pheromones and the heuristic information
- virtual void [computeProbas](#) ()
compute the probabilities for each decision of the artificial ants
- int [randomChoice](#) (int pos)
make a random choice based on _probas for a character in a given position
- void [evaporatePheromone](#) ()
evaporate the artificial pheromones
- virtual void [depositPheromones](#) ([Solution](#) &ant)
deposit pheromones
- virtual void [initPheromones](#) ()
init the pheromone matrix

Protected Attributes

- [Instance](#) & **_instance**
- std::vector< std::vector< double > > **_pheromones**
- std::vector< std::vector< double > > **_probas**
- double **_alpha**
- double **_beta**
- double **_rho**
- int **_nAnts**
- int **_nItMax**
- double **_initPheromone**
- std::vector< [Solution](#) > **_population**

4.2.1 Detailed Description

base class for ant colony metaheuristics for the Closest String Problem

4.2.2 Constructor & Destructor Documentation

4.2.2.1 AntColony::AntColony ([Instance](#) & *instance*)

constructor

Parameters

<i>instance</i>	instance of the CSP problem
-----------------	-----------------------------

4.2.3 Member Function Documentation

4.2.3.1 `virtual void AntColony::buildSolution (Solution & solution)` `[protected]`, `[virtual]`

build a new solution based on the pheromones and the heuristic information

Parameters

<i>solution</i>	the solution built
-----------------	--------------------

Reimplemented in [ACS](#).

4.2.3.2 `virtual void AntColony::depositPheromones (Solution & ant)` `[protected]`, `[virtual]`

deposit pheromones

Parameters

<i>ant</i>	the ant which deposits the pheromones
------------	---------------------------------------

Reimplemented in [MaxMin](#), and [ACS](#).

4.2.3.3 `int AntColony::randomChoice (int pos)` `[protected]`

make a random choice based on `_probas` for a character in a given position

Parameters

<i>pos</i>	the position to make a choice in
------------	----------------------------------

Returns

the choice made (character)

4.2.3.4 `virtual void AntColony::solve (Solution & best)` `[virtual]`

solve the CSP problem

Parameters

<i>best</i>	the best solution found
-------------	-------------------------

Reimplemented in [MaxMin](#), and [ACS](#).

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

- [src/AntColony.hpp](#)

4.3 Instance Class Reference

manage an instance of the csp problem

```
#include <Instance.hpp>
```

Public Member Functions

- [Instance](#) ()
default constructor
- bool [load](#) (std::string fileName)
load a instance of the CSP problem
- void [display](#) ()
display the instance
- int [stringLength](#) ()
return the length of the strings
- int [nString](#) ()
number of strings of the problem
- int [nChar](#) ()
number of characters of the alphabet
- char [getIndexChar](#) (int ind)
get the character represented by the index index
- std::vector< int > & [getString](#) (int ind)
access a string
- double [greedyScore](#) (int pos, int ch)
returns the greedy score of a character for a given position

4.3.1 Detailed Description

manage an instance of the csp problem

4.3.2 Member Function Documentation

4.3.2.1 char Instance::getIndexChar (int ind)

get the character represented by the index index

Parameters

<i>ind</i>	the index that represents the character in the problem
------------	--

Returns

the character

4.3.2.2 `std::vector<int>& Instance::getString (int ind)`

access a string

Parameters

<i>ind</i>	the index of the string
------------	-------------------------

Returns

a reference toward a string

4.3.2.3 `double Instance::greedyScore (int pos, int ch)`

returns the greedy score of a character for a given position

Parameters

<i>pos</i>	the position in the strings
<i>ch</i>	the character

4.3.2.4 `bool Instance::load (std::string fileName)`

load a instance of the CSP problem

Parameters

<i>fileName</i>	the name of the file
-----------------	----------------------

Returns

true iff the instance has been opened without troubles

4.3.2.5 `int Instance::nChar ()`

number of characters of the alphabet

Returns

the number of character

4.3.2.6 `int Instance::nString ()`

number of strings of the problem

Returns

number of strings of the problem

4.3.2.7 `int Instance::stringLength ()`

return the length of the strings

Returns

the length of the strings

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

- [src/Instance.hpp](#)

4.4 MaxMin Class Reference

class [MaxMin](#): Max Min ant colony optimization metaheuristic for the Closes String Problem

```
#include <MaxMin.hpp>
```

Inheritance diagram for MaxMin:

Collaboration diagram for MaxMin:

Public Member Functions

- [MaxMin](#) ([Instance](#) &instance)
costructor
- virtual void [solve](#) ([Solution](#) &best)
solve the CSP problem

Protected Member Functions

- virtual void `initPheromones` ()
init the pheromone matrix
- virtual void `depositPheromones` (`Solution` &`ant`)
deposit pheromones
- virtual void `updateBounds` (`Solution` &`best`)
update pheromone bounds bestAnt the best solution so far
- virtual void `checkBounds` ()
check if the pheromones satisfy the bounds

Additional Inherited Members

4.4.1 Detailed Description

class `MaxMin`: Max Min ant colony optimization metaheuristic for the Closes String Problem

4.4.2 Constructor & Destructor Documentation

4.4.2.1 `MaxMin::MaxMin (Instance & instance)`

costructor

Parameters

<i>instance</i>	instance of the Closest String Problem
-----------------	--

4.4.3 Member Function Documentation

4.4.3.1 `virtual void MaxMin::depositPheromones (Solution & ant)` `[protected]`, `[virtual]`

deposit pheromones

Parameters

<i>ant</i>	the ant which deposits the pheromones
------------	---------------------------------------

Reimplemented from `AntColony`.

4.4.3.2 `virtual void MaxMin::solve (Solution & best)` `[virtual]`

solve the CSP problem

Parameters

<i>best</i>	the best solution found
-------------	-------------------------

Reimplemented from [AntColony](#).

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

- [src/MaxMin.hpp](#)

4.5 Utils::Parameters Class Reference

parameters of the program

```
#include <Utils.hpp>
```

Public Attributes

- `std::string` **instance**
- `int` **nAnt**
- `int` **nItMax**
- `double` **alpha**
- `double` **beta**
- `double` **rho**
- `double` **a**
- `double` **convRate**
- `double` **q0**
- `double` **initPheromone**

4.5.1 Detailed Description

parameters of the program

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

- [src/Utils.hpp](#)

4.6 Solution Class Reference

class [Solution](#) representing a solution of the Closest String Problem

```
#include <Solution.hpp>
```

Public Member Functions

- [Solution](#) ([Instance](#) &instance)
constructor
- void [display](#) ()
display the solution
- void [setChar](#) (int pos, int character)
set the character of the solution in position pos
- void [setCharUpdate](#) (int pos, int character)
set the character of the solution in position pos and update the cost directly (for the local search)
- int [getChar](#) (int pos)
return a character at a given position
- int [cost](#) ()
return the cost of the solution
- [Solution](#) & [operator=](#) (const [Solution](#) &solution)
copy operator
- void [generateGreedy](#) ()
generate a greedy solution
- void [generateRandom](#) ()
generate a random solution
- void [localSearch](#) ()
perform a local search on the solution

4.6.1 Detailed Description

class [Solution](#) representing a solution of the Closest String Problem

4.6.2 Constructor & Destructor Documentation

4.6.2.1 [Solution::Solution](#) ([Instance](#) & *instance*)

constructor

Parameters

<i>instance</i>	the instance of the CSP problem
-----------------	---------------------------------

4.6.3 Member Function Documentation

4.6.3.1 int [Solution::cost](#) ()

return the cost of the solution

Returns

the cost

4.6.3.2 int Solution::getChar (int *pos*)

return a character at a given position

Parameters

<i>pos</i>	the position
------------	--------------

Returns

the character at position *pos*

4.6.3.3 Solution& Solution::operator= (const Solution & *solution*)

copy operator

Parameters

<i>solution</i>	the solution to copy
-----------------	----------------------

4.6.3.4 void Solution::setChar (int *pos*, int *character*)

set the character of the solution in position *pos*

Parameters

<i>pos</i>	the position
<i>character</i>	the character

4.6.3.5 void Solution::setCharUpdate (int *pos*, int *character*)

set the character of the solution in position *pos* and update the cost directly (for the local search)

Parameters

<i>pos</i>	the position
<i>character</i>	the character

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

- [src/Solution.hpp](#)

4.7 Utils Class Reference

definition of the class [Utils](#) containing the utilites functions

```
#include <Utils.hpp>
```

Classes

- class [Parameters](#)
parameters of the program

Static Public Member Functions

- static double [randomNumber](#) ()
return a random number between 0 and 1 (one excluded)

4.7.1 Detailed Description

definition of the class [Utils](#) containing the utilites functions

4.7.2 Member Function Documentation

4.7.2.1 static double [Utils::randomNumber](#) () [static]

return a random number between 0 and 1 (one excluded)

Returns

the random number

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

- src/[Utils.hpp](#)

Chapter 5

File Documentation

5.1 src/ACS.hpp File Reference

```
#include "AntColony.hpp"
```

Include dependency graph for ACS.hpp:

5.2 src/AntColony.hpp File Reference

definition of the virtual class [AntColony](#)

```
#include "Solution.hpp"
#include <vector>
```

Include dependency graph for AntColony.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class [AntColony](#)
base class for ant colony metaheuristics for the Closest String Problem

5.2.1 Detailed Description

definition of the virtual class [AntColony](#)

5.3 src/Instance.hpp File Reference

```
#include <string>
#include <vector>
#include <map>
```

Include dependency graph for Instance.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class [Instance](#)
manage an instance of the csp problem

5.4 src/MaxMin.hpp File Reference

definition of a class [MaxMin](#)

```
#include "AntColony.hpp"
```

Include dependency graph for MaxMin.hpp:

Classes

- class [MaxMin](#)
class [MaxMin](#): Max Min ant colony optimization metaheuristic for the Closes String Problem

5.4.1 Detailed Description

definition of a class [MaxMin](#)

5.5 src/Solution.hpp File Reference

definition of a class [Solution](#)

```
#include "Instance.hpp"
```

Include dependency graph for Solution.hpp: This graph shows which files directly or indirectly include this file:

Classes

- class [Solution](#)
class [Solution](#) representing a solution of the Closest String Problem

5.5.1 Detailed Description

definition of a class [Solution](#)

5.6 src/Utils.hpp File Reference

utilities for the arificial ant framework

```
#include <string>
```

Include dependency graph for Utils.hpp:

Classes

- class [Utils](#)
definition of the class [Utils](#) containing the utilites functions
- class [Utils::Parameters](#)
parameters of the program

5.6.1 Detailed Description

utilities for the arificial ant framework

Index

- ACS, [7](#)
 - buildSolution, [7](#)
 - depositPheromones, [8](#)
 - exploitationChoice, [8](#)
 - solve, [8](#)
- AntColony, [8](#)
 - AntColony, [9](#)
 - buildSolution, [10](#)
 - depositPheromones, [10](#)
 - randomChoice, [10](#)
 - solve, [10](#)
- buildSolution
 - ACS, [7](#)
 - AntColony, [10](#)
- cost
 - Solution, [16](#)
- depositPheromones
 - ACS, [8](#)
 - AntColony, [10](#)
 - MaxMin, [14](#)
- exploitationChoice
 - ACS, [8](#)
- getChar
 - Solution, [16](#)
- getIndexChar
 - Instance, [11](#)
- getString
 - Instance, [12](#)
- greedyScore
 - Instance, [12](#)
- Instance, [11](#)
 - getIndexChar, [11](#)
 - getString, [12](#)
 - greedyScore, [12](#)
 - load, [12](#)
 - nChar, [12](#)
 - nString, [13](#)
 - stringLength, [13](#)
- load
 - Instance, [12](#)
- MaxMin, [13](#)
 - depositPheromones, [14](#)
 - MaxMin, [14](#)
- solve, [14](#)
- nChar
 - Instance, [12](#)
- nString
 - Instance, [13](#)
- operator=
 - Solution, [17](#)
- randomChoice
 - AntColony, [10](#)
- randomNumber
 - Utils, [18](#)
- setChar
 - Solution, [17](#)
- setCharUpdate
 - Solution, [17](#)
- Solution, [15](#)
 - cost, [16](#)
 - getChar, [16](#)
 - operator=, [17](#)
 - setChar, [17](#)
 - setCharUpdate, [17](#)
 - Solution, [16](#)
- solve
 - ACS, [8](#)
 - AntColony, [10](#)
 - MaxMin, [14](#)
- src/ACS.hpp, [19](#)
- src/AntColony.hpp, [19](#)
- src/Instance.hpp, [19](#)
- src/MaxMin.hpp, [20](#)
- src/Solution.hpp, [20](#)
- src/Utils.hpp, [20](#)
- stringLength
 - Instance, [13](#)
- Utils, [17](#)
 - randomNumber, [18](#)
- Utils::Parameters, [15](#)