Table of contents

Abstract

This cookbook is intended to illustrate a Library that offers a solution for Data Structures and Algorithms with high performance.

In this cookbook, we provide two main recipes. The first recipe aims at illustrating

how to install and configure **Aleph-w Library**. The second one describes how

to add the **Aleph-w Library** to a QtProject.

Keywords

Aleph-w,C++, **Data structures, Algorithms.**

Contributors

Authors	Reviewers
Angelica Amaya Johanna Raimundo Ronald Sulbaran	Angelica Amaya Johanna Raimundo Ronald Sulbaran

Releases

Releases	Date	Author(s)	Description	Status*
0.1		Angelica Amaya,Johanna Raimundo,Ronald Sulbaan	Abstract and recipes proposal to reviewers	Reviewed
0.2		Angelica Amaya,Johanna Raimundo,Ronald Sulbaan	Introduction and definitions Recipe 1 Recipe 2	Reviewed Reviewed Reviewed
1.0		Angelica Amaya,Johanna Raimundo,Ronald Sulbaan	Public release of V 1.0	

* Status: Started, ToReview, Reviewed, Revised, Published

Introduction

Software engineering is the study of ways in which to create large and complex computer applications and that generally involve many programmers and designers. At the heart of software engineering is with the overall design of the applications and on the creation of a design that is based on the needs and requirements of end users. While software engineering involves the full life cycle of a software project, is includes many different components - specification, requirements gathering, design, verification, coding, testing, quality assurance, user acceptance testing, production, and ongoing maintenance.

Having an in-depth understanding on every component of software engineering is not mandatory, however, it is important to understand that the subject of data structures and algorithms is concerned with the coding phase. The use of data structures and algorithms is the nuts-and-blots used by programmers to store and manipulate data.

Aleph-w is a library of abstract data types and algorithms around a wide range of fundamental data structures and computing issues

The remainder of this cookbook is organized as follows. First, some definitions are intended to provide a basic understanding of Data Structures, Algorithms and the used tools. Then, several recipes will show you how to implement and used Aleph-w library.
This cookbook is dedicated to Linux users.

Definitions

Data Structures and Algorithms

A data structure is an arrangement of data in a computer's memory or even disk storage. An example of several common data structures are arrays, linked lists, queues, stacks, binary trees, and hash tables. Algorithms, on the other hand, are used to manipulate the data contained in these data structures as in searching and sorting.

Many algorithms apply directly to a specific data structures. When working with certain data structures you need to know how to insert new data, search for a specified item, and deleting a specific item.

Commonly used algorithms include are useful for:

- Searching for a particular data item (or record).
- Sorting the data. There are many ways to sort data. Simple sorting, Advanced sorting
- Iterating through all the items in a data structure. (Visiting each item in turn so as to display it or perform some other action on these items)

Aleph-w Library

Aleph-w is a library of data structures and algorithms implemented in and for C++.

It contains the main data structures: single and double lists, special arrays (dynamic and of bits), many types of trees: multitrees, binary search trees, avl, red-black, randomized, treaps, splay and other interesting types.

Several types of hash tables are implemented: separated chaining, open adressing with linear probing and double function hash; this last one has a garbage colector mechanism that allows to free deleted entries. Dynamic linear hash tables are implemented too.

The main algorithms on graphs are implemented in an easy way and with good performance features. Network Flow (maximum and min cost), cut points, topological sort, spanning trees, min paths. etc. The graphs are generic in the sense that they can contain data independent of the algorithm.

Several containers of the C++ STL library are implemented with Aleph-w. Their performance is better than GNU implementation

Clang

Clang is a compiler front end for the C, C++, Objective-C and Objective-C++ programming languages. It uses LLVM as its back end and has been part of the LLVM release cycle since LLVM 2.6.

It is designed to offer a complete replacement to the GNU Compiler Collection (GCC). It is open-source, developed by Apple; other companies such as Google are involved. Its source code is available under the University of Illinois/NCSA License.

The Clang project includes the Clang front end and the Clang static analyzer and several code analysis tools.

Recipes

Recipe1: Installing Aleph-w Library

In order to carry out this recipe you will need:

- The g++ compiler >= 4.8.
- The clang system >= 3.4.
- The Aleph-w library >= 1.5a.2 that can be downloaded from [[1] Aleph-w library]

Step1: Installing g++-4.8

Firstly, you need to open a terminal and type:

sudo apt-get install g++-4.8 and press enter key.

Finally, run g++--version and to confirm that the g++-4.8 is

installed.

```
ronalanto03@ronalanto03:~$ g++ --version
g++ (Ubuntu 4.8.1-2ubuntu1~12.04) 4.8.1
Copyright (C) 2013 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

It's important that you see the number 4.8 for the g++ version number.

Step2: Installing the clang system

First: you need to open a terminal and type:

sudo gedit /etc/apt/sources.list (You can use gedit or any other text editor**)**

Second: add the folowing lines to the sources.list file depending on your linux system:

For Debian:

wheezy (Debian stable) - Last update : Sat, 28 Dec 2013 15:42:47 UTC / Revision: 198086

```
deb http://llvm.org/apt/wheezy/ llvm-toolchain-wheezy main
deb-src http://llvm.org/apt/wheezy/ llvm-toolchain-wheezy main
```

sid (unstable) - Last update : Mon, 23 Jun 2014 11:34:16 UTC / Revision: 211489

```
deb http://llvm.org/apt/unstable/ llvm-toolchain main
deb-src http://llvm.org/apt/unstable/ llvm-toolchain main
# 3.4
deb http://llvm.org/apt/unstable/ llvm-toolchain-3.4 main
deb-src http://llvm.org/apt/unstable/ llvm-toolchain-3.4 main
```

For Ubuntu:

Precise (12.04) - Last update : Tue, 17 Jun 2014 00:39:17 UTC / Revision: 211056

```
deb http://llvm.org/apt/precise/ llvm-toolchain-precise main
deb-src http://llvm.org/apt/precise/ llvm-toolchain-precise main
# 3.4
deb http://llvm.org/apt/precise/ llvm-toolchain-precise-3.4 main
deb-src http://llvm.org/apt/precise/ llvm-toolchain-precise-3.4 main
# Common
deb http://ppa.launchpad.net/ubuntu-toolchain-r/test/ubuntu precise main
```

Quantal (12.10) - Last update : Mon, 16 Jun 2014 13:24:06 UTC / Revision: 198077

```
deb http://llvm.org/apt/quantal/ llvm-toolchain-quantal main
deb-src http://llvm.org/apt/quantal/ llvm-toolchain-quantal main
# 3.4
deb http://llvm.org/apt/quantal/ llvm-toolchain-quantal-3.4 main
deb-src http://llvm.org/apt/quantal/ llvm-toolchain-quantal-3.4 main
# Common
deb http://ppa.launchpad.net/ubuntu-toolchain-r/test/ubuntu quantal main
```

Raring (13.04) - Last update : Tue, 17 Jun 2014 05:01:23 UTC / Revision: 211088

```
deb http://llvm.org/apt/raring/ llvm-toolchain-raring main
deb-src http://llvm.org/apt/raring/ llvm-toolchain-raring main
# 3.4
deb http://llvm.org/apt/raring/ llvm-toolchain-raring-3.4 main
deb-src http://llvm.org/apt/raring/ llvm-toolchain-raring-3.4 main
# Common
deb http://ppa.launchpad.net/ubuntu-toolchain-r/test/ubuntu raring main
```

Saucy (13.10) - Last update : Mon, 23 Jun 2014 17:56:31 UTC / Revision: 211499

```
deb http://llvm.org/apt/saucy/ llvm-toolchain-saucy main
deb-src http://llvm.org/apt/saucy/ llvm-toolchain-saucy main
# 3.4
deb http://llvm.org/apt/saucy/ llvm-toolchain-saucy-3.4 main
deb-src http://llvm.org/apt/saucy/ llvm-toolchain-saucy-3.4 main
```

Trusty (14.04) - Last update : Mon, 23 Jun 2014 16:59:06 UTC / Revision: 211499

```
deb http://llvm.org/apt/trusty/ llvm-toolchain-trusty main
deb-src http://llvm.org/apt/trusty/ llvm-toolchain-trusty main
# 3.4
deb http://llvm.org/apt/trusty/ llvm-toolchain-trusty-3.4 main
```

```
deb-src http://llvm.org/apt/trusty/ llvm-toolchain-trusty-3.4 main
```

Third:

Install (stable branch)

- To retrieve the archive signature:

  type in a terminal **sudo wget -O - http://llvm.org/apt/llvm-snapshot.gpg.key|sudo apt-key add -**

- To install just clang, lldb and llvm (3.4 release):

  type in a terminal **sudo apt-get install clang-3.4 lldb-3.4 llvm-3.4**

Step3: Installing the aleph library

Firstly, you need to open the Imakefile file that is inside the folder of the Aleph-w library and just delete or comment lines if you wish to use gnu or clang suite, in this case we are going to use clang suite:

```
# just delete or comment these four lines if you wish to use gnu
# compiler suite

# adjust this path to your clang path
CLANGPATH=
CC =clang
#CC = /home/lrleon/GCC/bin/gcc
AS = llvm-as-3.4
AR = llvm-ar-3.4 clq
LD = llvm-link-3.4 -nostdlib

# now comment clang++ line and uncomment c++ line. Flag for standard could
# change according to compiler version. This flag is for gcc 4.6.3. On
# 4.7.x should work -std=c++11
CXX =clang++
#CXX = /home/lrleon/GCC/bin/c++

# Perform make OPTIONS="-Ofast -D_REENTRANT -DWITHOUT_NANA" if you want
# to generate optimized libAleph.a. If you use clang compiler, then you
# may be intested in to use -O4; in this case, be sure that llvm be
# installed with gold linker and the gold plugin. Be careful with some
# gsl parts and the -Ofast option in clang; they do not compile!
#
# In Aleph-w clang generated code is between 5-10% faster than gcc

HLIST=   tpl_dynSetTree.H tpl_arrayHeap.H tpl_binHeap.H  \
   tpl_dynBinHeap.H generate_tree.H generate_graph.H  \
```

Plain Text ▾ Tab Width: 3 ▾ Ln 3, Col 1 INS

Second:

Type in a terminal : **sudo find / -name 'c++config.h' -exec echo {} \;** and press enter.

```
ronalanto03@ronalanto03:~/Downloads/alephClang (copy)$ sudo find / -name 'c++config.h' -exec echo {} \;
/usr/include/i386-linux-gnu/c++/4.8/bits/c++config.h
/usr/include/i386-linux-gnu/c++/4.8/64/bits/c++config.h
/usr/include/i386-linux-gnu/c++/4.7/bits/c++config.h
/usr/include/i386-linux-gnu/c++/4.7/64/bits/c++config.h
/usr/include/c++/4.4/i686-linux-gnu/bits/c++config.h
/usr/include/c++/4.4/i686-linux-gnu/64/bits/c++config.h
/usr/include/c++/4.6/i686-linux-gnu/bits/c++config.h
/usr/include/c++/4.6/i686-linux-gnu/64/bits/c++config.h
/opt/Xilinx/12.4/ISE_DS/EDK/gnu/microblaze/lin/include/c++/4.1.2/microblaze-xilinx-elf/bits/c++config.h
/opt/Xilinx/12.4/ISE_DS/EDK/gnu/powerpc-eabi/lin/include/c++/4.1.1/powerpc-eabi/bits/c++config.h
/opt/cadence/soc_encounter/tools.lnx86/systemc/gcc/4.1/install/include/c++/4.1.2/i686-pc-linux-gnu/bits/c++config.h
/opt/cadence/soc_encounter/tools.lnx86/systemc/gcc/4.1-x86_64/install/include/c++/4.1.2/x86_64-redhat-linux/bits/c++config.h
ronalanto03@ronalanto03:~/Downloads/alephClang (copy)$ sudo find / -name 'c++config.h' -exec echo {} \;
```

Select the path that contains **c++/4.8**

"/usr/include/i386-linux-gnu/c++/4.8"

Third:

look for the line that contains **"INCLUDES = -I./"** in the Imakefile file and change it by "**INCLUDES = -I./ -I/usr/include/i386-linux-gnu/c++/4.8"** obtained before.

Fourth:

Open a terminal cd to aleph directory and run: **xmkmf**

```
ronalanto03@ronalanto03:~/Downloads/alephClang (copy)$ xmkmf
mv -f Makefile Makefile.bak
imake -DUseInstalled -I/usr/lib/X11/config
```

run **make libAleph.a** and wait for the compilation.

```
ronalanto03@ronalanto03:~/Downloads/alephClang (copy)$ make libAleph.a
clang++ -std=c++11 -I./ -I/usr/include/i386-linux-gnu/c++/4.8 -DDEBUG -O0 -g3 -pthread -DWITH_NANA -DMESSAGES -Wall -Wextra -Wcast-align -Wno-sign-compare -Wno
-write-strings -Wno-parentheses -D_GLIBCXX__PTHREADS -D__extern_always_inline=inline   -c -o eepicgeom.o eepicgeom.C
clang++ -std=c++11 -I./ -I/usr/include/i386-linux-gnu/c++/4.8 -DDEBUG -O0 -g3 -pthread -DWITH_NANA -DMESSAGES -Wall -Wextra -Wcast-align -Wno-sign-compare -Wno
-write-strings -Wno-parentheses -D_GLIBCXX__PTHREADS -D__extern_always_inline=inline   -c -o point.o point.C
clang++ -std=c++11 -I./ -I/usr/include/i386-linux-gnu/c++/4.8 -DDEBUG -O0 -g3 -pthread -DWITH_NANA -DMESSAGES -Wall -Wextra -Wcast-align -Wno-sign-compare -Wno
-write-strings -Wno-parentheses -D_GLIBCXX__PTHREADS -D__extern_always_inline=inline   -c -o euclidian-graph-common.o euclidian-graph-common.C
clang++ -std=c++11 -I./ -I/usr/include/i386-linux-gnu/c++/4.8 -DDEBUG -O0 -g3 -pthread -DWITH_NANA -DMESSAGES -Wall -Wextra -Wcast-align -Wno-sign-compare -Wno
-write-strings -Wno-parentheses -D_GLIBCXX__PTHREADS -D__extern_always_inline=inline   -c -o ahUtils.o ahUtils.C
clang++ -std=c++11 -I./ -I/usr/include/i386-linux-gnu/c++/4.8 -DDEBUG -O0 -g3 -pthread -DWITH_NANA -DMESSAGES -Wall -Wextra -Wcast-align -Wno-sign-compare -Wno
-write-strings -Wno-parentheses -D_GLIBCXX__PTHREADS -D__extern_always_inline=inline   -c -o hash-fct.o hash-fct.C
```

Recipe2: Adding Aleph-w Library to a QtProject

In order to carry out this recipe you will need:

- The QtCreator and QtLibraries.

- The Aleph-w library >=1.5a.2.

Step1: Open a qtcreator project

Open qtcreator, go to File and create or open a new project.

Step2: Edit the .pro file

Open the *.pro file and add the following lines:

```
QMAKE_CXXFLAGS += -std=c++11
CONFIG += c++11
INCLUDEPATH+=$(AlephPath)

LIBS =-L$(AlephPath)\
-lnana -lc -lm -lgsl -lgslcblas -lgmp -lmpfr -lasprintf
```

```
#-------------------------------------------------
#
# Project created by QtCreator 2014-01-21T13:16:33
#
#-------------------------------------------------

QMAKE_CXXFLAGS += -std=c++11
CONFIG += c++11
INCLUDEPATH+=/home/ronalanto03/Downloads/aleph

LIBS =-L/home/ronalanto03/Downloads/aleph\
-lnana -lc -lm -lgsl -lgslcblas -lgmp -lmpfr -lasprintf

QT       += core

QT       -= gui

TARGET = AlgoritmoSimilitud
CONFIG   += console
CONFIG   -= app_bundle

TEMPLATE = app


SOURCES += main.cpp \
    usersnetwork.cpp \
    client.cpp \
    server.cpp \
    socketsLibrary/Socket_Servidor.cpp \
    socketsLibrary/Socket_Cliente.cpp \
    socketsLibrary/Socket.cpp
```

Recommended documentation

[1] Aleph-w library

http://sourceforge.net/projects/aleph-w/

Feedback

Comments and suggestions: angelica.guldris@gmail.com, johap7@hotmail.com, ronalanto03@hotmail.com