UNIVERSIDAD DE LA LAGUNA

DEPARTAMENTO DE ESTADÍSTICA,
INVESTIGACIÓN OPERATIVA Y COMPUTACIÓN

SEMINARIO

LEDA: Una librería de clases en C++ aplicable a problemas de optimización

Marcos Colebrook Santamaría

La Laguna, 22 de abril de 1999

Contenidos

- Introducción a LEDA
- Breve historia de LEDA
- ¿Quién usa actualmente LEDA?
- Software compatible con LEDA
- Instalación de LEDA
- Introducción al C++
- Tipos (clases) implementados en LEDA
- ¿Cómo puedo programar en LEDA?
- Documentación y manual disponibles
- LEDA en Internet
- Demo en LEDA

Introducción a LEDA

- LEDA: "*Library of Efficient Datatypes and Algorithms*".
- Filosofía: "*To provide a library of efficient data types and algorithms of combinatorial computing that is attractive to non-experts that would not be used otherwise".*
- Contiene implementaciones eficientes de cada tipo de datos.
- Proporciona un tipo "grafo" muy cómodo. Ofrece las iteraciones clásicas como "`para todos los nodos v del grafo G`" o "`para todos los vecinos w de v hacer`", además de la inserción o eliminación de vértices y aristas.
- LEDA está implementado en una librería de clases en C++. Puede ser usado prácticamente con cualquier compilador: `g++`, `CC`, `xlC`, `cxx`, `Borland C++`, `MSVC++` y `Watcom`.
- LEDA no es de domino público, pero puede usarse gratuitamente para fines docentes e investigadores. También existe una versión comercial.

Breve historia de LEDA

- LEDA nace en otoño de 1988.
- Estuvieron 6 meses con las especificaciones y la elección del lenguaje. Probaron SmallTalk, Modula, Ada, Eiffel y C++, decantándose por C++.
- En la 2ª mitad de 1989 y durante 1990, Stefan Näher implementó un primera versión de LEDA (version 1.0).
- Luego siguieron la versión 2.0 (junio 1992), la 3.0 (febrero 1993), la 3.1 (enero 1995), y así hasta la 4.2 (actual).
- En cada nueva versión se añadían nuevos tipos de datos y se arreglaban los "bugs" de la versión anterior, además de aprovechar las ventajas que proporcionaba el lenguaje C++.
- Responsables de LEDA: Kurt Melhorn, Stefan Näher y Christian Uhrig.

¿Quién usa actualmente LEDA?

- Chevron Petroleum (USA)
- Comptel (Finlandia)
- Digital Equipment (USA)
- Daimler Benz (Alemania)
- E-Plus Mobilfunk (Alemania)
- France Telecom (Francia)
- Lufthansa Systems (Alemania)
- Ford Motor Company (USA)
- MCI (USA)
- Siemens AG (Alemania)
- Silicon Graphics (USA)
- Sony Corporation (USA)
- Sun Microsystems (USA)

Software compatible con LEDA

- **ABACUS:** A Branch and Cut Framework for Polyhedral Optimization.
- **AGD:** Algorithms for Graph Drawing.
- **CGAL:** Computational Geometry Algorithms Library.
- **GDToolkit**: Graph Drawing Toolkit.
- **Proyectos:**
  - Asstuce, ASSUME, CRSM, DISCCOVER, ESMI, GeoMAMOS, GeoRouting, LoLA, Map Labeling, ML & KBANN, MLC++, MOSES, OCEAN, OLT, OptiLRV, PlaNet, RoLoPro, TERA, TGM, VISPAK, etc.

Introducción al C++ (I)

- **C++ es una ampliación del lenguaje C que permite la *programación orientada a objetos*.**
- **Básicamente, una clase es una estructura que engloba DATOS y PROCEDIMIENTOS.**
- **Una instancia (variable) de una clase dada es un OBJETO.**
- **Por tanto, el objeto contendrá:**
  - **Datos (privados): corresponden a las características que definen ese objeto.**
  - **Procedimientos (públicos): indican qué es lo que yo puedo hacer con ese objeto.**

Introducción al C++ (II)

- Por ejemplo: la clase “array” de números enteros podría definirse como

```
class array
{ int Longitud;        // Indica la longitud del array.
  int *Ptr;            // El array de enteros.
  ...
public:
  ...
  int ValorMaximo(); // Busca el máximo valor.
  void Ordenar();    // Ordenar todos los componentes.
};

array A;
...
A.ordenar();
int Maximo = A.ValorMaximo();
...
```

Instalación de LEDA

- Descargar de la dirección Internet el soft según el sistema.
- Hay disponible actualmente LEDA para:
  - Sun, HP, IBM, linux: para sus compiladores CC, g++, egcs y xlC
  - PC: para los compiladores de Borland, Microsoft y Watcom.
- Descomprimir el soft (.tgz / .zip).
- Seguir las instrucciones del fichero INSTALL.
- En UNIX:
  - `setenv LEDAROOT <direc. LEDA> / export LEDAROOT=<direc. LEDA>`
  - Incluir los directorios de búsqueda de las librerías:

    ```
    g++ -I$LEDAROOT/incl -L$LEDAROOT ...
    ```
- Básicamente sólo son importantes dos cosas:
  - Los ficheros .h (include): localizados en `$LEDAROOT/incl`
  - Las librerias estáticas/dinámicas (.a / .sl / .so): están en `$LEDAROOT`

Tipos (clases) implementados en LEDA

- Tipos de datos simples
  - `string, date, file, etc.`
- Tipos número y álgebra lineal
  - `integer, rational, vector, matrix, etc.`
- Tipos de datos básicos
  - `array, stack, queue, set, list, etc.`
- Diccionarios (hashing)
  - `dictionary, map, sortseq, etc.`
- Colas de prioridad
  - `p_queue, b_priority_queue, etc.`
- Grafos
  - `graph, ugraph, GRAPH, UGRAPH, node_array, node_set, node_list, edge_array, edge_set, graph_gen, markov_chain, etc.`
- Algoritmos sobre grafos
  - `shortest_path, max_flow, min_cost_flow, mc_mathing, min_span, etc.`
- Iteradores sobre grafos
  - `NodeIt, EdgeIt, OutAdjIt, InAdjIt, etc.`
- Geometría en 2D
  - `point, segment, line, circle, etc.`
- Tipos avanzados para 2D
  - `d2_dictionary, point_set, interval_set, segment_set, etc.`
- Geometría en 3D
  - `d3_point, d3_plane, d3_hull, etc.`
- Gráficos
  - color, window, menu, GraphWin, etc.

¿Cómo puedo programar en LEDA? (I)

```
$ cat > array.C
#include <iostream.h>
#include <LEDA/array.h>

#define N 10

main()
{ array<int> A(N);

  for (int i = 0; i < N; i++)
     A[i] = i + 1;
  A.print(); newline;

  A.permute(); A.print(); newline;
  A.sort(); A.print(); newline;
}
^D
$ g++ -I/opt/LEDA/incl -L/opt/LEDA
  array.C -o array -lW -lP -lG -lL -lm
```

```
$ array
1 2 3 4 5 6 7 8 9 10
7 9 5 8 10 6 3 2 1 4
1 2 3 4 5 6 7 8 9 10
```

¿Cómo puedo programar en LEDA? (II)

```
$ cat > grafo.C
#include <iostream.h>
#include <LEDA/ugraph.h>
#include <LEDA/graph_gen.h>

main()
{ ugraph G;

  complete_ugraph(G, 5);
  cout << G << endl;
}
^D
$ g++ -I/opt/LEDA/incl -L/opt/LEDA
  grafo.C -o grafo -lW -lP -lG -lL -lm
```

```
$ grafo
LEDA.GRAPH
void
void
5
|{}|
|{}|
|{}|
|{}|
|{}|
10
1 2 0 |{}|
1 3 0 |{}|
1 4 0 |{}|
1 5 0 |{}|
2 3 0 |{}|
2 4 0 |{}|
2 5 0 |{}|
3 4 0 |{}|
3 5 0 |{}|
4 5 0 |{}|
```

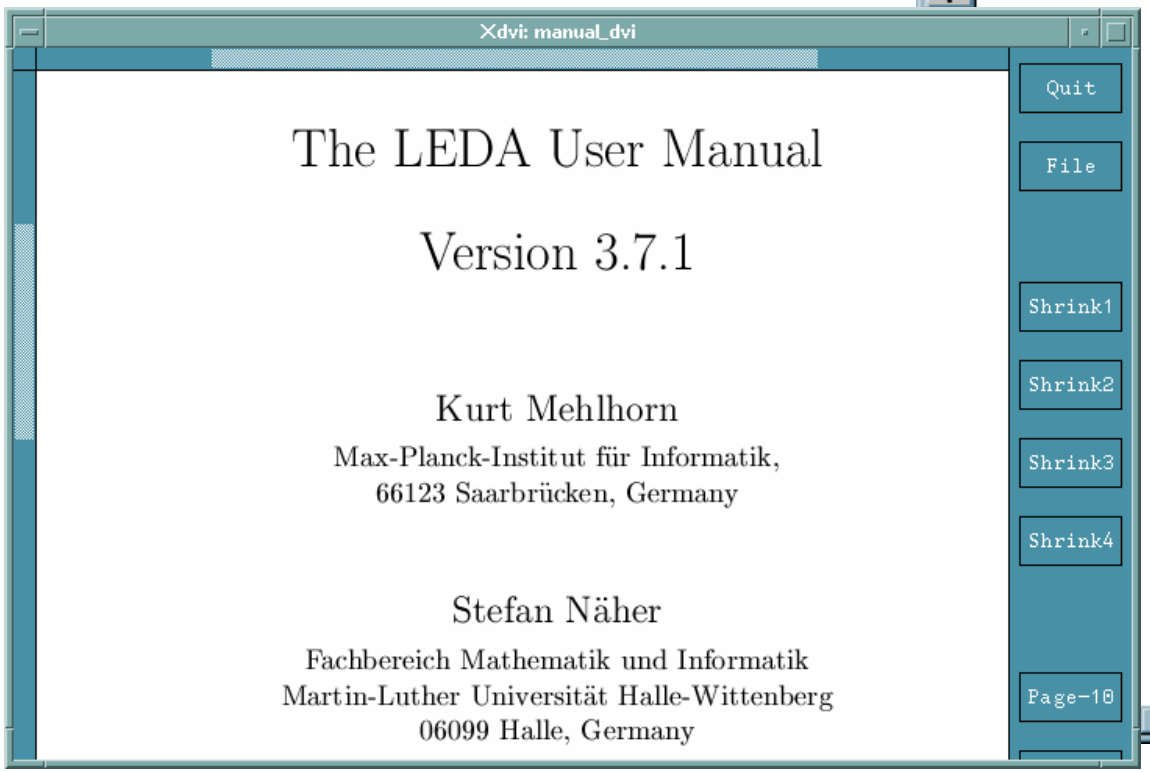
¿Cómo puedo programar en LEDA? (III)

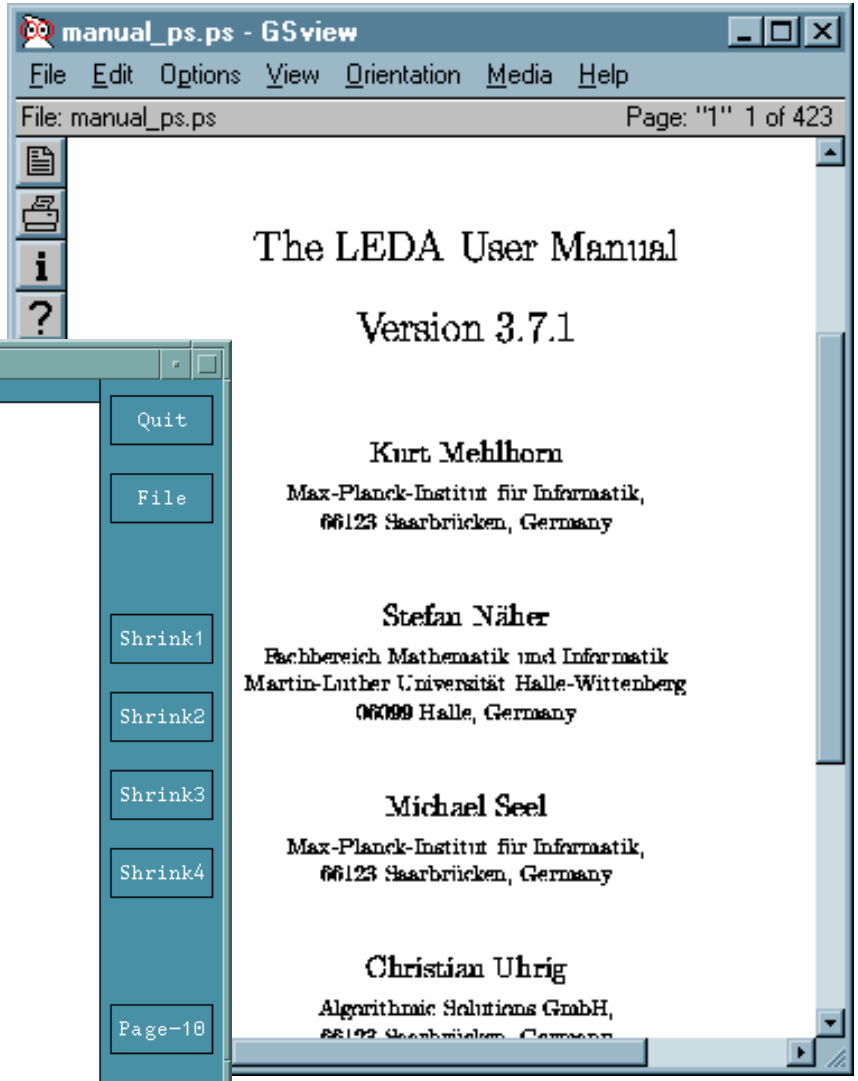
```
$ cat > grafo2.C
#include <iostream.h>
#include <LEDA/ugraph.h>
#include <LEDA/graph_gen.h>
#include <LEDA/min_span.h>

main()
{ UGRAPH<int, int> G;

  complete_ugraph(G, 5);
  cout << G << endl;
  cout << SPANNING_TREE(G) << endl;
}
^D
$ g++ -I/opt/LEDA/incl -L/opt/LEDA
  grafo2.C -o grafo2 -lW -lP -lG -lL -
  lm
```

```
$ grafo2
LEDA.GRAPH
int
int
5
|{0}|
|{0}|
|{0}|
|{0}|
|{0}|
10
1 2 0 |{0}|
1 3 0 |{0}|
1 4 0 |{0}|
...
4 5 0 |{0}|

0x40013d6c 0x40013d98 0x40013dc4
0x40013df0
```

Documentación y manuales disponibles

- **Manual en HTML en INTERNET.**
- **Manual en HTML local.**
- **Manual en PostScript.**
- **Manual en DVI.**





LEDA en Internet

- **URL oficial:**

  http://www.mpi-sb.mpg.de/LEDA

- **News de LEDA (en desuso):**

  comp.lang.c++.leda

- **Lista de distribución (LEDA Discussion):**

  http://postino.mpi-sb.mpg.de/cgi-bin/lyris.pl?enter=leda-discussion&text_mode=0&lang=english

- **Lista de distribución para docencia (LEDA Teaching):**

  http://postino.mpi-sb.mpg.de/cgi-bin/lyris.pl?enter=leda-teaching&text_mode=0&lang=english