

INSTRUCCIONES:

1. Como incluir problemas nuevos: Ej.: las mareas de venecia.

Pasos:

1. Modificar “/util/ParamsObject.java” en xmlresults y xmlparams, afecta a todos los proyectos (Server, ROS, Worker y PServer)

P.Ej. ROS:

Línea 490: Introduccion XML

```

} else if ((Language.equalsIgnoreCase("c++")) &&
(Type.equalsIgnoreCase("simpleea"))){
    this.set("execution","make SEQ");
    this.set("compilation","make MainSeq");
    this.set("Online","");
    this.set("comment","Problema de las mareas");
}

```

Línea 610 (Entradas al algoritmo)

```

} else if ((Language.equalsIgnoreCase("c++")) &&
(Type.equalsIgnoreCase("simpleea"))){
params = "\t\t<number_of_genes>          "+NumberGenes+"          </number_of_genes>\n"+
"\t\t<length_of_gene>          "+LengthGene+"          </length_of_gene>\n"+
"\t\t<population_size>          "+PopulationSize+"          </population_size>\n"+
"\t\t<fitness_function>          "+Fitness_Function+"          </fitness_function>\n";
}

```

Línea 675 (Resultados)

```

} else if ((Language.equalsIgnoreCase("c++")) &&
(Type.equalsIgnoreCase("simpleea"))){
results = "\t\t<individual>          "+Individual+"          </individual>\n"+
"\t\t<fitness>          "+Fitness+"          </fitness>\n"+
"\t\t<best_fitness>          "+Best_Fitness+"          </best_fitness>\n"+
"\t\t<worst_fitness>          "+Worst_Fitness+"          </worst_fitness>\n"+
"\t\t<generation>          "+Generation+"          </generation>\n"+
"\t\t<avg_fitness>          "+Avg_Fitness+"          </avg_fitness>\n"+
"\t\t<stddev_fitness>          "+Stddev_Fitness+"          </stddev_fitness>\n"+
"\t\t<computation_time>          "+Computation_Time+"          </computation_time>\n"+
"\t\t<error>          "+Error+"          </error>\n";
}

```

* Todos los parámetros que se envían y reciben deben tener un orden específico que concuerde con los presentados por pantalla y en DTD.

2. Modificar el fichero “typeop_file.txt” de primaryServer, y añadir un tipo de problema más que se denomine “xxxxxx” con una IP del servidor que lo vaya a solucionar (127.0.0.1 sería el local).
3. Modificar “clientWindow.java” en línea 194 para poner en el combo (listbox) de lenguajes de la aplicación, el lenguaje necesario.
P.Ej: C++ para SimpleEA (antes únicamente estaba previsto el Java).

```

} else if ((Language.equalsIgnoreCase("c++")) &&
(Type.equalsIgnoreCase("simpleea"))){
results = "\t\t<individual>          "+Individual+"          </individual>\n"+
"\t\t<fitness>          "+Fitness+"          </fitness>\n"+
"\t\t<best_fitness>          "+Best_Fitness+"          </best_fitness>\n"+
"\t\t<worst_fitness>          "+Worst_Fitness+"          </worst_fitness>\n"+
"\t\t<generation>          "+Generation+"          </generation>\n"+
"\t\t<avg_fitness>          "+Avg_Fitness+"          </avg_fitness>\n"+
"\t\t<stddev_fitness>          "+Stddev_Fitness+"          </stddev_fitness>\n"+
"\t\t<computation_time>          "+Computation_Time+"          </computation_time>\n"+
"\t\t<error>          "+Error+"          </error>\n";
}

```

4. Modificar “**VisualOpt.java**” para crear (new) de “VP_TipoAlg” en línea 141.

```

}
else if ((cw.getTypeSelect().trim().equalsIgnoreCase("simpleea")) &&
(cw.getLanguage().trim().equalsIgnoreCase("C++"))){

vp_simpleea = new VP_simpleea(xml_file,translate);
jtp.addTab(cw.getTypeSelect().toUpperCase()+"
"+translate.getText(20),(VP_simpleea)vp_simpleea);

```

Y poner también declaración:

```
static VP_simpleea vp_simpleea = null; (Línea 41)
```

5. Crear “**VP_TipoAlg.java**” si no existe y hacerlo a partir de uno existente. Modificar su aspecto gráfico, añadiendo o quitando parámetros de entrada y/o salida.

```

dis = new DataInputStream(new FileInputStream(xml_file));
String actual = "";
while ((current = dis.readLine())!=null){
    if ((current.startsWith("\t\t<") || (current.startsWith("\t\t\t<"))){
        actual =
current.substring((current.indexOf('<')+1),current.indexOf('>'));
        if (current.endsWith("/>"))
            actual =
current.substring((current.indexOf('<')+1),current.indexOf(' '));
        System.out.println("ACTUAL..." +actual);
        if (actual.equalsIgnoreCase(""))
            break;
        else if (actual.equalsIgnoreCase("number_of_genes")){
            vf_ng = new VisualField(translate.getText(81));
            vf_ng.setValue("10");

            InputPanel.add(vf_ng);
            incount += 1;
        }else if (actual.equalsIgnoreCase("length_of_gene")){
            vf_gl = new VisualField(translate.getText(82));
            vf_gl.setValue("2");

            InputPanel.add(vf_gl);
            incount += 1;
        }else if (actual.equalsIgnoreCase("population_size")){
            vf_ps = new VisualField(translate.getText(83));
            vf_ps.setValue("100");

            InputPanel.add(vf_ps);
            incount += 1;
        }else if (actual.equalsIgnoreCase("population_number")){
            vf_np = new VisualField(translate.getText(84));
            InputPanel.add(vf_np);
            incount += 1;
        }else if (actual.equalsIgnoreCase("fitness_function")){
            //vp_ef.setLayout(new GridLayout(1,3));
            button_ef.addActionListener( alfc );
            vp_ef.add(new JLabel(translate.getText(85)));
            vp_ef.add(funcEval);
            vp_ef.add(button_ef);
            InputPanel.add(vp_ef);
            incount += 1;
        }else if (actual.equalsIgnoreCase("individual")){ // Abs Error Training
            vf_i = new VisualField(translate.getText(86));
            vf_i.disable();
            OutputPanel.add(vf_i);
            outcount += 1;
        }else if (actual.equalsIgnoreCase("fitness")){ // Abs. Error Test
            vf_f = new VisualField(translate.getText(87));
            vf_f.disable();
            OutputPanel.add(vf_f);
            outcount += 1;
        }else if (actual.equalsIgnoreCase("best_fitness")){ // RMS Training
            vf_cpt = new VisualField(translate.getText(88));
            vf_cpt.disable();
            OutputPanel.add(vf_cpt);
            outcount += 1;
        }else if (actual.equalsIgnoreCase("worst_fitness")){ // Recognize TEST
            vf_error = new VisualField(translate.getText(89));
            vf_error.disable();
            OutputPanel.add(vf_error);
            outcount += 1;
        }else if (actual.equalsIgnoreCase("generation")){ //Recognize Training
            vf_gen = new VisualField(translate.getText(90));
            vf_gen.disable();
            OutputPanel.add(vf_gen);
            outcount += 1;
        }else if (actual.equalsIgnoreCase("avg_fitness")){ // RMS Test
            vf_avg = new VisualField(translate.getText(91));
            vf_avg.disable();
            OutputPanel.add(vf_avg);
            outcount += 1;
        }else if (actual.equalsIgnoreCase("Stddev_Fitness")){ // Rule Number
            vf_std = new VisualField(translate.getText(92));
            vf_std.disable();
            OutputPanel.add(vf_std);
            outcount += 1;
        }
    }
}

```

6. Modificar el fichero de idiomas: **“idiom.txt”** para que se traduzcan correctamente las palabras añadidas al formulario (van referenciadas por número de posición).

Modificar **“VP_TipoAlg.java”**: 271 para ejecución del programa EXE (**envío / recepción**(punto 9))

```
static public void getActionRun(){
    int [] arrayInt = new int[1];
    long init_time, end_time;

    po.set("client_number","1");
    po.set("Language",po.get("language").trim());
    po.set("number_of_genes",vf_ng.getValue());
    po.set("length_of_gene",vf_gl.getValue());
    po.set("population_size",vf_ps.getValue());
    po.set("fitness_function",funcEval.getText());
    poToFile(xml_file);
}
```

7. Modificar **“Worker.java”** (Línea 110) para incluir según algoritmo que llega por XML, que WRAPPER instanciar (**“wrapper_TipoAlgoritmo.java”**)

```
}else if ((type_of_algor.equalsIgnoreCase("simpleea")) &&
(langu.equalsIgnoreCase("C++"))){
    wp = new Wrapper_simpleea2(xml_file);}
}
```

8. Modificar WORKER: crear archivo **“wrapper_TipoAlgoritmo.java”** a partir de uno existente, y modificar en los siguientes puntos:
 - a. Rutas/Nombres de Archivos
 - b. Formato y estructura del fichero de configuración que tomara el programa como entrada. (entrada.txt)
 - c. Ejecución del programa (ubicación, parámetros, etc..)
 - d. Recepción del archivo de texto de resultados.txt (en nuestro caso, parámetros separados por \$).

El envío y recepción de parámetros debe tener concordancia con los recibidos/enviados por XML, de lo contrario habrá una excepción del SAX.

9. Modificar por último el **“VP_TipoAlg.java”** de ROS para la recepción correcta y en orden de los datos, y su impresión por pantalla. Para ello es posible tener que realizar declaraciones como:

```
static VisualField vf_avg;//
static VisualField vf_std;//
static VisualField vf_ct;//
static VisualField vf_gen;//
static VisualField vf_error;// Campos de respuesta especificos para mareas

(.....)

//          -***** Queda interpretar los datos que vienen de vuelta *****
vf_i.setValue(po_res.get("individual").trim());
vf_f.setValue(po_res.get("fitness").trim());
vf_cpt.setValue(po_res.get("best_fitness").trim());
vf_error.setValue(po_res.get("worst_fitness").trim());
vf_gen.setValue(po_res.get("generation").trim());
vf_avg.setValue(po_res.get("avg_fitness").trim());
vf_std.setValue(po_res.get("Stddev_Fitness").trim());
vf_cmt.setValue(po_res.get("computation_time").trim()+
"+translate.getText(76));
```

Observaciones:

1. Es necesario devolver un Null en el parámetro ERROR para que funcione la vuelta de parámetros correctamente.
2. Todas las modificaciones de ejemplo realizadas en “**VP_TipoAlg.java**”, EVANNAI las ha realizado sobre “**VP_SimpleEA.java**”.