ECJ GENETIC

A Java-based Evolutionary Computation and Genetic Programming Research System

Néstor Arias
(naariasm@unal.edu.co)

CONTENIDO

- Qué es ECJ?
- Características Generales.
- Características de GP.
- Proceso Evolutivo General.
- Clases principales.
- Ejemplo.
- Conclusiones.

Qué es ECJ?

- Es una biblioteca de clases escrita en Java. Fue diseñada para ser flexible, casi todas sus clases y parámetros se determinan en tiempo de ejecución mediante archivos de parámetros.

Características Generales

- Tiene funciones de logging y checkpoint independientes de la plataforma.
- Archivos de parámetros jerárquicos.
- Multihilo.
- Buenos generadores de números aleatorios (Mersenne Twister).

Características Generales

- Soporte para varias técnicas evolutivas.
- Soporte para integración con el paquete Teambots, para hacer la evaluación.
- Características de computación evolutiva
- Modelos de islas asincrónicos sobre TCP/IP.

Características Generales

- Algoritmos genéticos / Programación Genética de estado estable o generacionales, con o sin elitismo.
- Estrategias evolutivas: evolución: mu, lambda y mu + lambda.
- Arquitectura de reproducción (breeding) bastante flexible.

Características Generales

- Gran número de operadores de selección.
- Soporte para múltiples subpoblaciones y especies.
- Intercambios entre subpoblaciones.
- Soporte para serialización de poblaciones en archivos.

Características de los individuos

- Genes de longitud fija o variable.
- Representaciones arbitrarias.
- Siete dominios de problemas soportados por defecto (sum, rosenbrock, sphere, step, noisy-quartic)
- Siete dominios de problemas soportados por defecto(hormigas, regresión, multiplexación, paridad, dos cajas, edge).

Características Generales

- Coevolución de población simple o múltiple.
- Optimización multiobjetivo SPEA2.
- Posibilidad de extender la librería para otros métodos de optimización multiobjetivo.

Características De Programación Genética

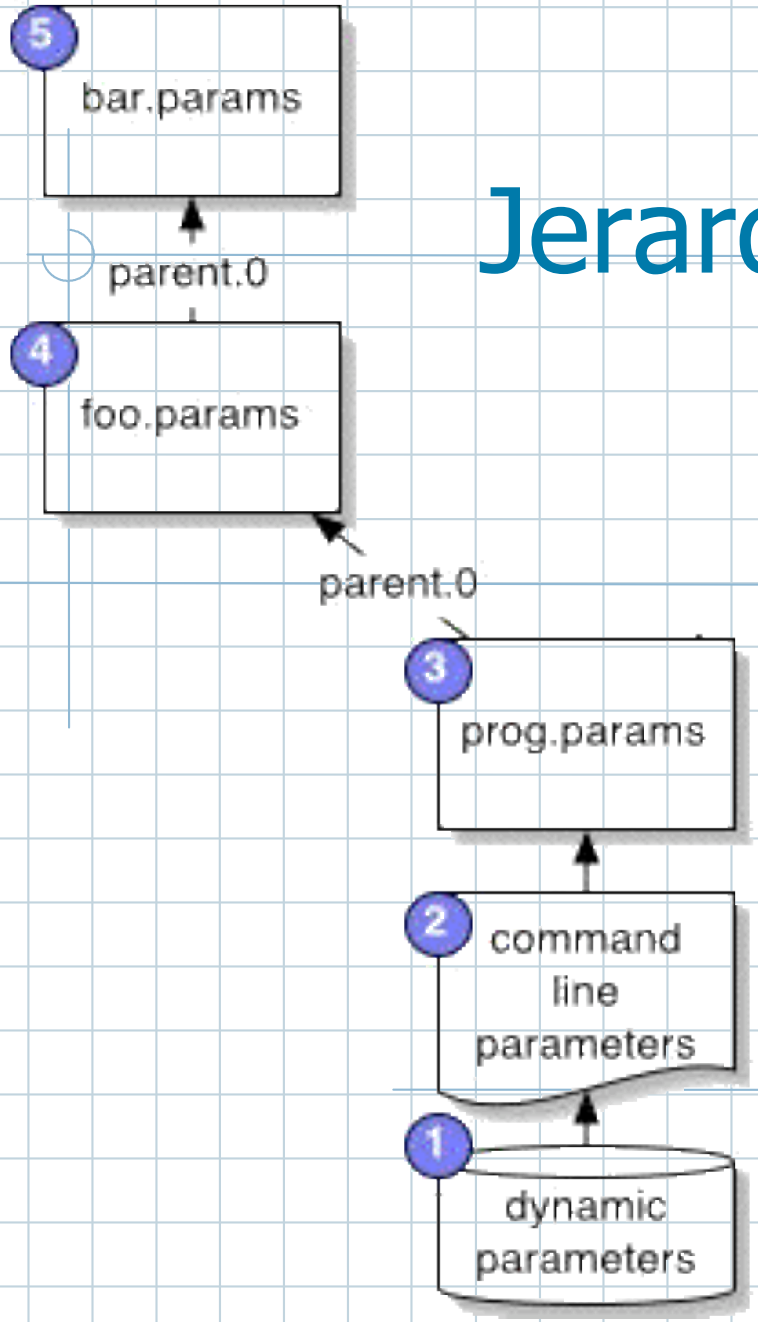
- Funciones definidas automáticamente y macros definidas automáticamente.
- Bosques de múltiples árboles.
- Seis algoritmos de creación de árboles.
- Gran número de operadores de reproducción.

Archivos de parámetros

- Guían la ejecución de ECJ.
- Son jerárquicos.
- Java ec.Evolve –file hola.params

```
pop.subpop.0.species.genome-size = 20
pop.subpop.0.species.crossover-type = two
pop.subpop.0.species.crossover-prob = 1.0
pop.subpop.0.species.mutation-prob = 0.01
```

Jerarquía de parámetros



Proceso Evolutivo

- La clase **Evolve** es el punto de entrada, es la que contiene el método main().

- Configura los generadores de números aleatorios, checkpoint, log, la db de parámetros y el objeto **EvolutionState**. Todo esto lo configura en base a un archivo de parámetros que da el usuario.

EvolutionState

- EvolutionState, es un Singleton, que contiene el estado completo del proceso evolutivo en un momento dado.
- Contiene los generadores de números aleatorios, las poblaciones, la base de datos de parámetros.

EvolutionState

- Es una clase abstracta, tiene el ciclo principal evolutivo, sus principales subclases son evolución de estado estable y generacional.
- EvolutionState contiene otros cinco objetos vitales en la evolución.

Initializer

- Es responsable de crear la población inicial. Puede leerla desde archivo o crearla aleatoriamente.

Evaluator

- Evalúa una población y le asigna un fitness.

Breeder

- Se encarga de crear la nueva población a partir de la antigua.

Exchanger

- Realiza intercambios entre subpoblaciones, posiblemente que están en diferentes máquinas.

Finisher

- Es una especie de destructor y es poco usado.

Statistics

- Se encarga de calcular y guardar las estadísticas.

Algoritmo general

Poblaciones e Individuos

- El objeto EvolutionState contiene un único objeto Population, el cual a su vez tiene un arreglo de subpoblaciones.
- Cada subpoblación contiene un arreglo de **Individual**s y un objeto **Species** al que estos pertenecen.

Individual

- Contiene un objeto de la clase Fitness, que representa el fitness del individuo.
- Las clases Species, individual y Fitness son abstractas.

Clase Species

- Representa a un grupo de individuos que tienen el mismo tipo de genes.
- Contiene un individuo prototipo que es usado para generar nuevos individuos.
- Contiene un BreedingPipeline que indica como será el proceso reproductivo.

BreedingPipeline

- Es el mecanismo básico para pasar de una generación a otra (Ej: selección, cruce, mutación).
- ECJ es bastante flexible en este aspecto y se puede personalizar bastante.

Clase Problem

- La clase **Evaluator** contiene un problema prototipo.

Ejemplo

- Para resolver un problema sencillo se debe extender la clase **Problem** e implementar la interface **SimpleProblemForm**.

- Se debe implementar el método **evaluate**() y el método **describe**().

Función Evaluate

```
public void evaluate(EvolutionState state,
                     Individual ind,
                     int threadnum)     {
    // si este individuo ya esta evaluado
    // retornar
    if (ind.evaluated == true)          {
        return;
    }
```

```
if (!(ind instanceof BitVectorIndividual))        {
            state.output.fatal("El individuo no es de
tipo BitVector");
            return;
        }
```

```
 BitVectorIndividual individual = (BitVectorIndividual)
ind;
        double fitness = fitness(individual);
        if (!(individual.fitness instanceof SimpleFitness))
{
            state.output.fatal("Horror, el fitness no es un
SimpleFitness");
            return;
    }
```

```
 SimpleFitness fitnessObj = (SimpleFitness)
individual.fitness;

        fitnessObj.setFitness(state, (float) fitness,
false);


        ind.evaluated = true;

}
```

Archivo de parámetros

```
state     = ec.simple.SimpleEvolutionState
pop       = ec.Population
init      = ec.simple.SimpleInitializer
finish    = ec.simple.SimpleFinisher
breed     = ec.simple.SimpleBreeder
eval      = ec.simple.SimpleEvaluator
stat      = ec.simple.SimpleShortStatistics
exch      = ec.simple.SimpleExchanger
```

Parámetros administrativos

generations = 200

quit-on-run-complete = true

gc = false

Aggressive = true

gc-modulo = 1

checkpoint = false

prefix = ec

checkpoint-modulo = 1

stat.file = $out.stat

Parámetros de la población

pop.subpops = 1

pop.subpop.0 = ec.Subpopulation

pop.subpop.0.size = 10

pop.subpop.0.duplicate-retries = 0

pop.subpop.0.fitness = ec.simple.SimpleFitness

pop.subpop.0.species = ec.vector.VectorSpecies

pop.subpop.0.species.ind = ec.vector.BitVectorIndividual

Más Parámetros

```
pop.subpop.0.species.genome-size = 20
pop.subpop.0.species.crossover-type = two
pop.subpop.0.species.crossover-prob = 1.0
pop.subpop.0.species.mutation-prob = 0.01
```

Breeding Pipeline

```
pop.subpop.0.species.pipe              =
   ec.vector.breed.VectorMutationPipeline
pop.subpop.0.species.pipe.source.0     =
   ec.vector.breed.VectorCrossoverPipeline
pop.subpop.0.species.pipe.source.0.source.0  =
   ec.select.TournamentSelection
pop.subpop.0.species.pipe.source.0.source.1  =
   ec.select.TournamentSelection
select.tournament.size     = 2
```

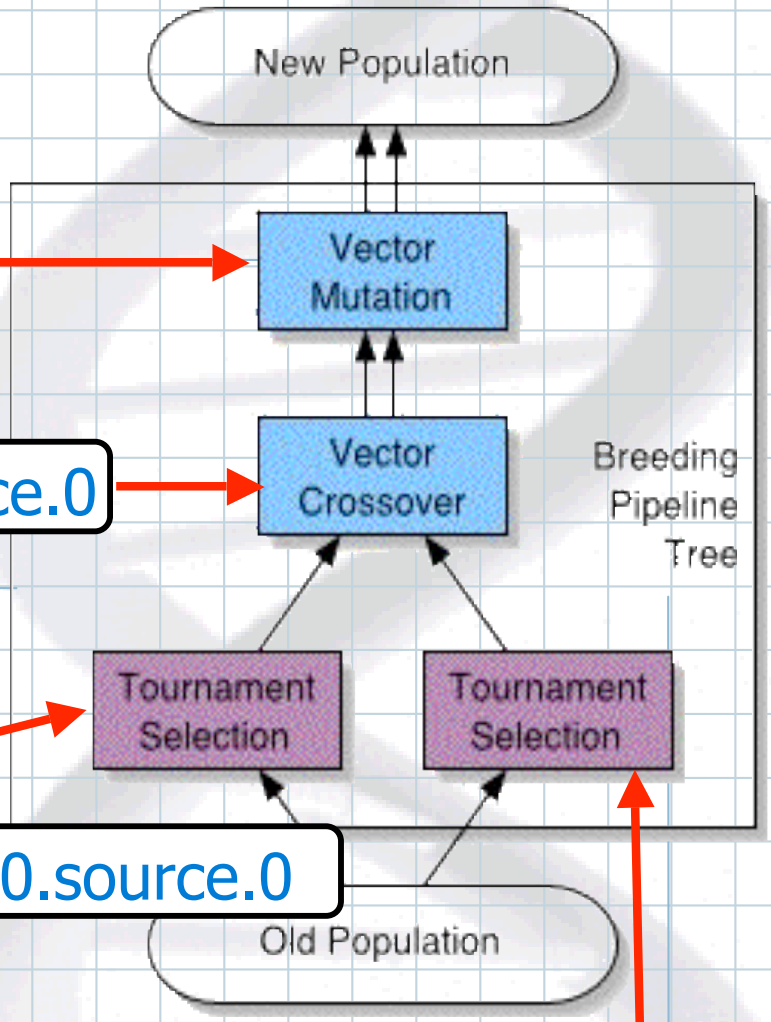
Breeding Pipeline

pop.subpop.0.species.pipe

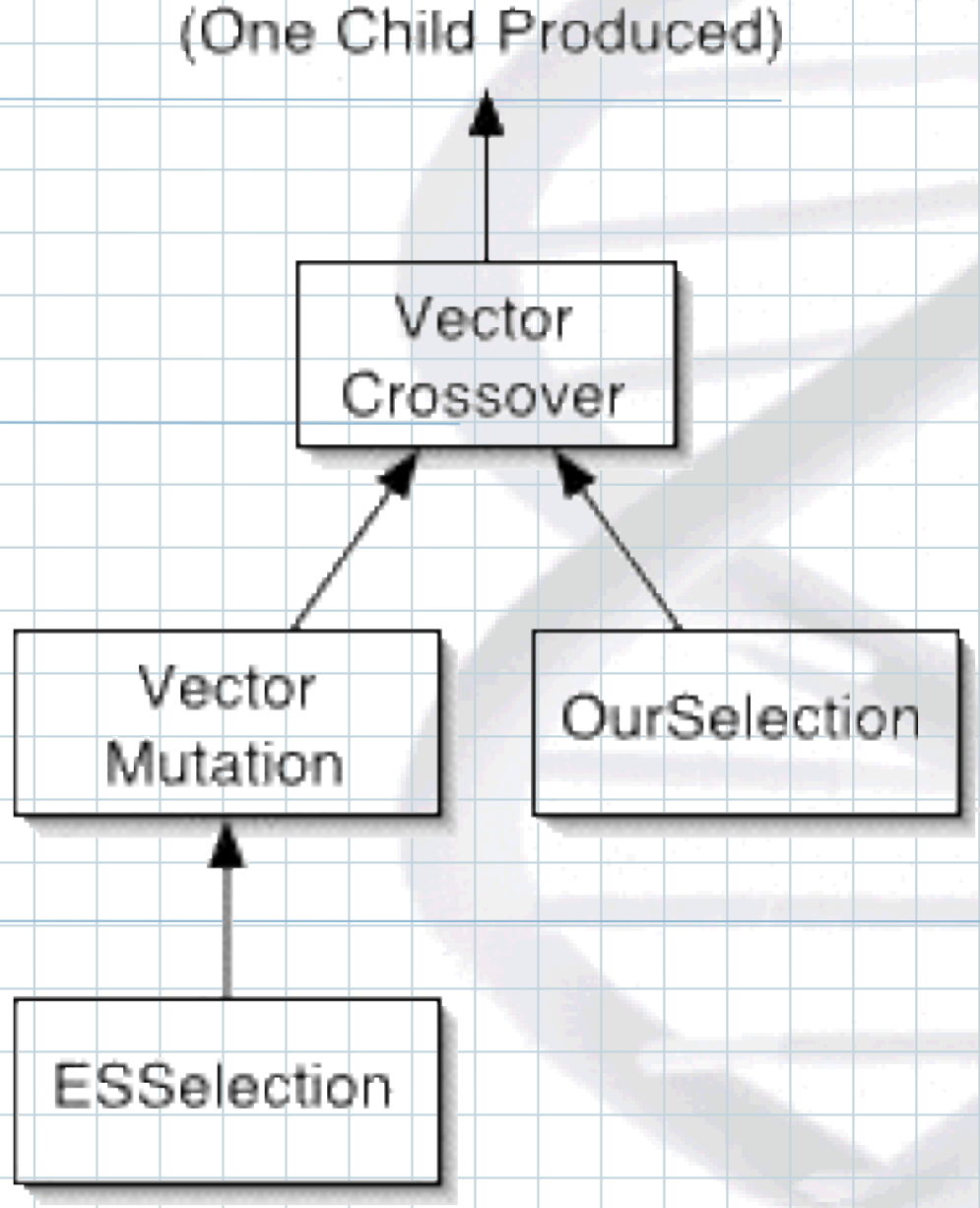
pop.subpop.0.species.pipe.source.0

pop.subpop.0.species.pipe.source.0.source.0

pop.subpop.0.species.pipe.source.0.source.1



Ejemplo de un Breeding Pipeline



Parámetros del problema

- eval.problem = co.edu.unal.compuEvolutiva.Knapsack01Problem
- eval.problem.file = problem0.txt

- breed.elites.0 = 1

Conclusiones

- Ventajas
  - Todas las que hereda de Java, como multiplataforma, multihilo, manejo de excepciones, garbage collection.
  - Gran Flexibilidad.
  - Varios problemas resueltos.

Conclusiones (Desventajas)

- Demasiada flexibilidad (Hay clase hasta para el fitness).
- Complejidad.
- En ocasiones puede cerrarse solo (cuando hay un error grave) lo que no lo hace propicio para aplicaciones WEB.
- No es totalmente transparente en cuanto a los hilos.
- Más que una biblioteca de clases es un programa de experimentación.
- No tiene visualizadores o wizards.

Otras desventajas

- Hay otros proyectos de computación evolutiva con Java que si están en sourceforge.net (uno de los más importantes sitios de proyectos open source), estos son JGAP y JAGA.