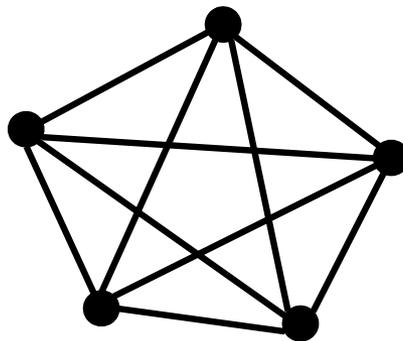
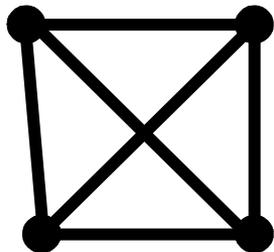


**Rozmiestnenie vrcholov kompletného grafu
metódou hillclimbingu s gaussovskou
mutáciou**

Evolučné algoritmy, cvičenie
Michal Moravčík
20.4.2006

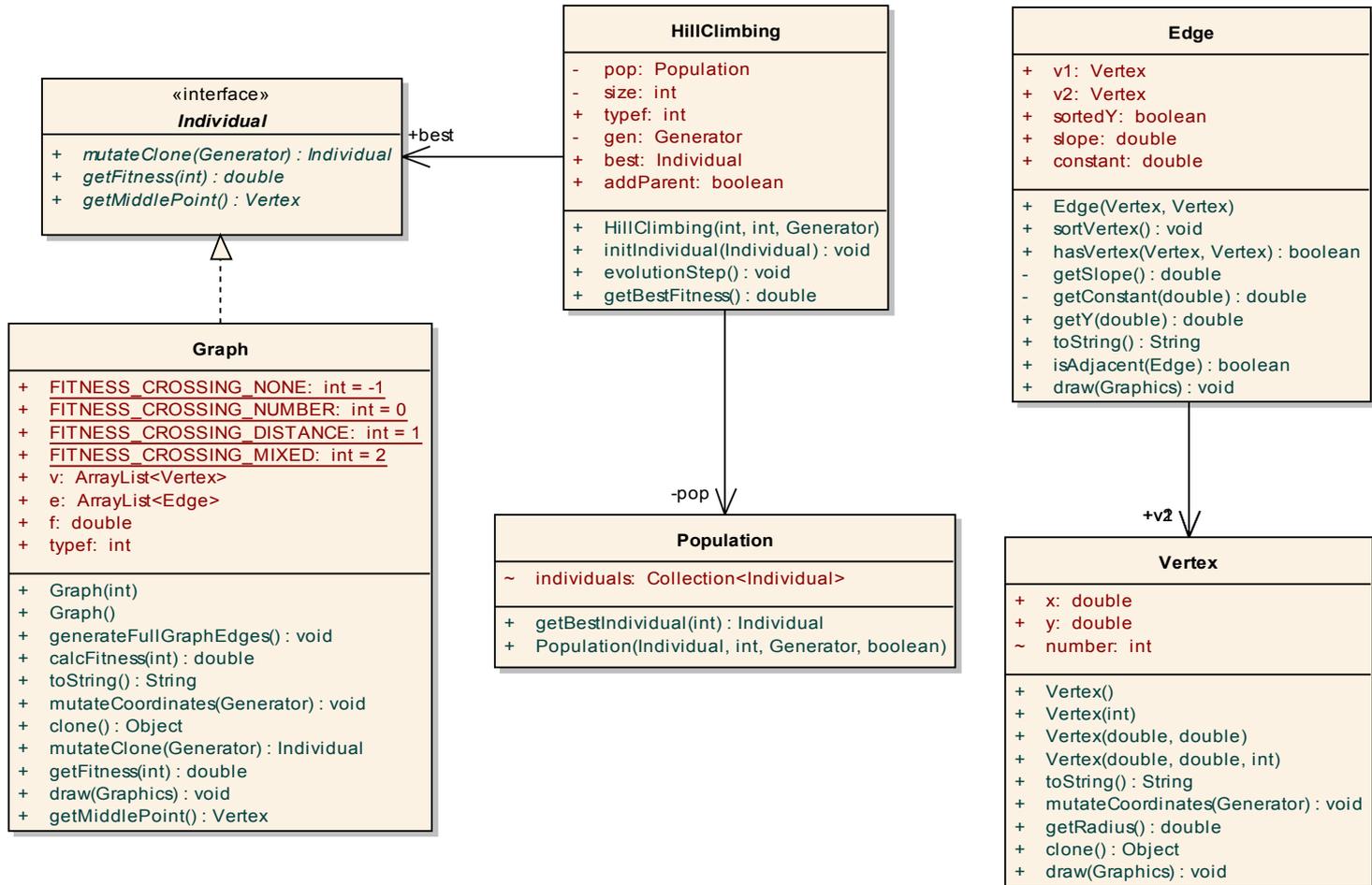
Zadanie problému



- Použite hillclimbing s gaussovskou mutáciou so sigmou 0,1 umiestnenia vrcholov na nájdenie rozmiestnenia vrcholov kompletného grafu pre $n=4$ a $n=5$ s čo najmenším počtom prekrížení.
- Počiatočné umiestnenia vrcholov by mali byť v štvorci o jednotkovej hrane, ale mutácie môžu vysunúť vrcholy mimo tohto štvorca. Vypočítajte priemer a smerodajnú odchýlku na počet pokusov k dosiahnutiu cieľa pre každý z grafov.

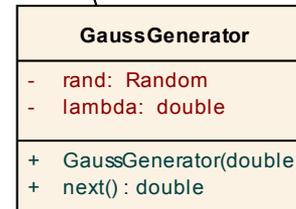
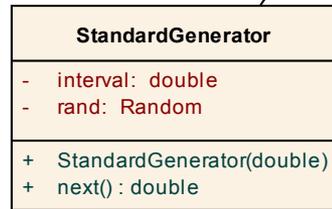
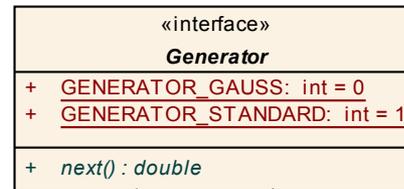
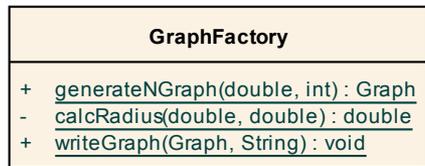
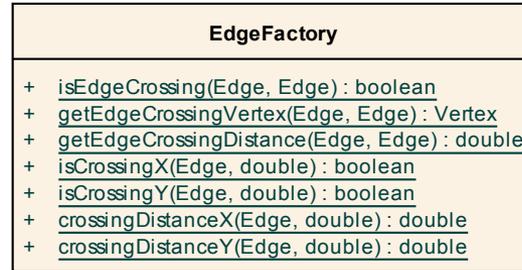
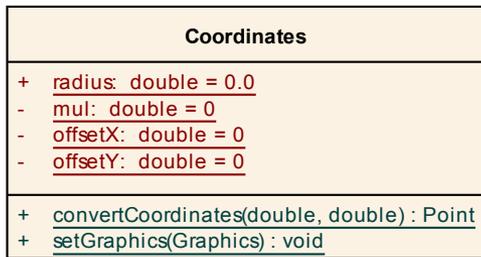
Dátový model

cd evolution



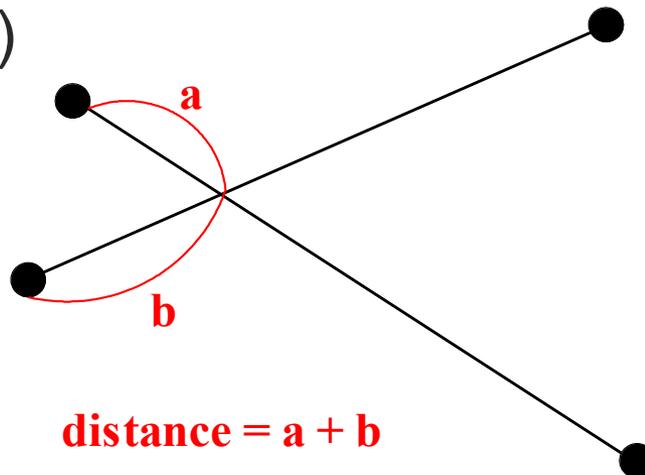
Dátový model (pokr.)

cd factory



Použité heuristiky pre fitness

- Hľadá sa minimum fitness
 - Počet prekrížených hrán (Number)
 - Súčet vzdialeností priesečníkov od koncov hrán (Distance)
 - Kombinácia = súčet prekrížení a vzdialeností (Mixed)



Rozpoznávání překřížení

```
public static boolean isEdgeCrossing(Edge e1, Edge e2) {
    if (e1.isAdjacent(e2)) return false;
    double a1 = e1.slope;
    double a2 = e2.slope;
    // e1 is vertical
    if (Double.isInfinite(a1) && !Double.isInfinite(a2)) {
        double yi = e2.getY(e1.v1.x);
        return (isCrossingY(e1, yi));
    }
    // e2 is vertical
    else if (!Double.isInfinite(a1) && Double.isInfinite(a2)) {
        double yi = e1.getY(e2.v1.x);
        return (isCrossingY(e2, yi));
    }
    // both e1 and e2 are vertical
    else if (Double.isInfinite(a1) && Double.isInfinite(a2))
        if (e1.v1.x == e2.v1.x) {
            return (isCrossingY(e1, e2.v1.y) || isCrossingY(e2, e1.v1.y));
        } else return false;
    // e1 and e2 are parallel
    else if (a1 == a2)
        if (e1.constant == e2.constant) { // same line
            return (isCrossingX(e1, e2.v1.x) || isCrossingX(e2, e1.v1.x));
        } else return false;
    else {
        double xi = (a1 * e1.v1.x - e1.v1.y - a2 * e2.v1.x + e2.v1.y) / (a1 - a2);
        return (isCrossingX(e1, xi) && isCrossingX(e2, xi));
    }
}
```

[Rozpoznávání překřížení (pokr.)]

```
public static boolean isCrossingX(Edge e, double xi) {  
    return (e.v1.x <= xi && xi <= e.v2.x);  
}
```

```
public static boolean isCrossingY(Edge e, double yi) {  
    if (e.sortedY)  
        return (e.v1.y <= yi && yi <= e.v2.y);  
    else  
        return (e.v2.y <= yi && yi <= e.v1.y);  
}
```

```
public boolean isAdjacent(Edge other) {  
    return (v1 == other.v1 || v1 == other.v2 ||  
            v2 == other.v1 || v2 == other.v2);  
}
```

[Evolučný algoritmus]

```
Individual graph = GraphFactory.generateNGraph(nSize);  
Generator generator = new GaussGenerator(lambda);  
HillClimbing hill = new HillClimbing(populationSize,  
    fitnessType, generator);
```

```
hill.initIndividual(graph);
```

```
while (true) {
```

```
    hill.evolutionStep(); pop = new Population(best, size, gen, addParent);  
    best = pop.getBestIndividual(typef);
```

```
    if (hill.getBestFitness() <= finalFitness || stop) {  
        break;
```

```
    }
```

```
}
```

Výsledky merania – 4 hrany

<i>population</i>	5	5	20	20	100	100
<i>target</i>	0	0	0	0	0	0
<i>generator</i>	gauss	standard	gauss	standard	gauss	standard
<i>generation</i>	15	44	20	23	5	218
	19	59	6	47	2	14
	11	203	15	65	37	27
	18	45	9	42	5	135
	19	93	5	71	2	23
	15	23	35	54	6	121
	14	37	7	21	7	49
	6	50	16	13	30	32
	11	28	6	25	9	84
	29	17	6	11	4	20
<i>average</i>	15.7	59.9	12.5	37.2	10.7	72.3
<i>deviation</i>	4.44	35.24	7.2	18.6	9.12	53.76

Výsledky merania – 5 hráč

<i>population</i>	5	5	20	20	100	100
<i>target</i>	1	0.01	1	0.01	1	0.01
<i>heuristics</i>	number	distance	number	distance	number	distance
<i>generation</i>	11	12	5	8	5	6
	22	8	12	4	7	5
	14	16	5	6	31	5
	24	9	23	6	15	7
	190	16	24	8	14	3
	11	12	56	6	9	15
	15	6	55	11	4	15
	16	6	13	11	35	3
	148	21	5	6	8	4
	13	13	21	8	11	8
<i>average</i>	46.4	11.9	21.9	7.4	13.9	7.1
<i>deviation</i>	49.04	3.72	14.08	1.8	7.88	3.34

[Výsledky merania – n-hrán]

n-size	6	7	8	9	10	11	12
crossing	3	9	19	36	62	102	153

[Ukážka]

