Evolutionary Computation

Algorithms + Data Structures = Programs (N.Wirth)

Genetic Algorithms + Data Structures = Evolution Programs

Evolution Programs can be perceived as a **generalization of genetic algorithms.**

•Classical genetic algorithms operate on fixed length binary strings, which need not be the case for evolution programs.

•Evolution programs usually incorporate a variety of genetic operators, wherever classical genetic algorithms use binary crossover and mutation.

Application Areas

- Search
- Optimization
- Machine Learning

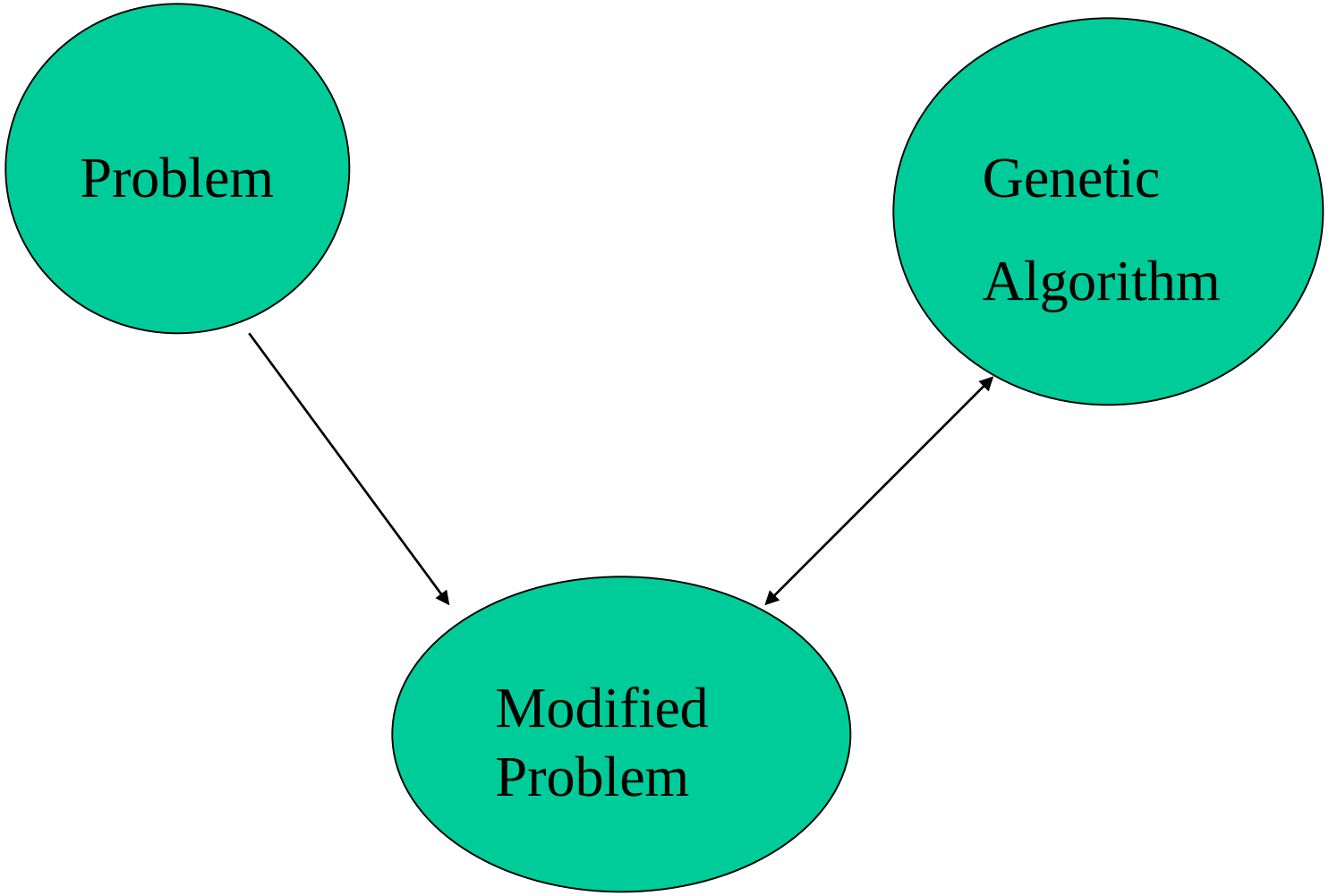
Evolutionary Computation

- Genetic algorithms
- Genetic Programming
- Evolution Strategies
- Evolutionary Programming

Structure of an Evolution Program

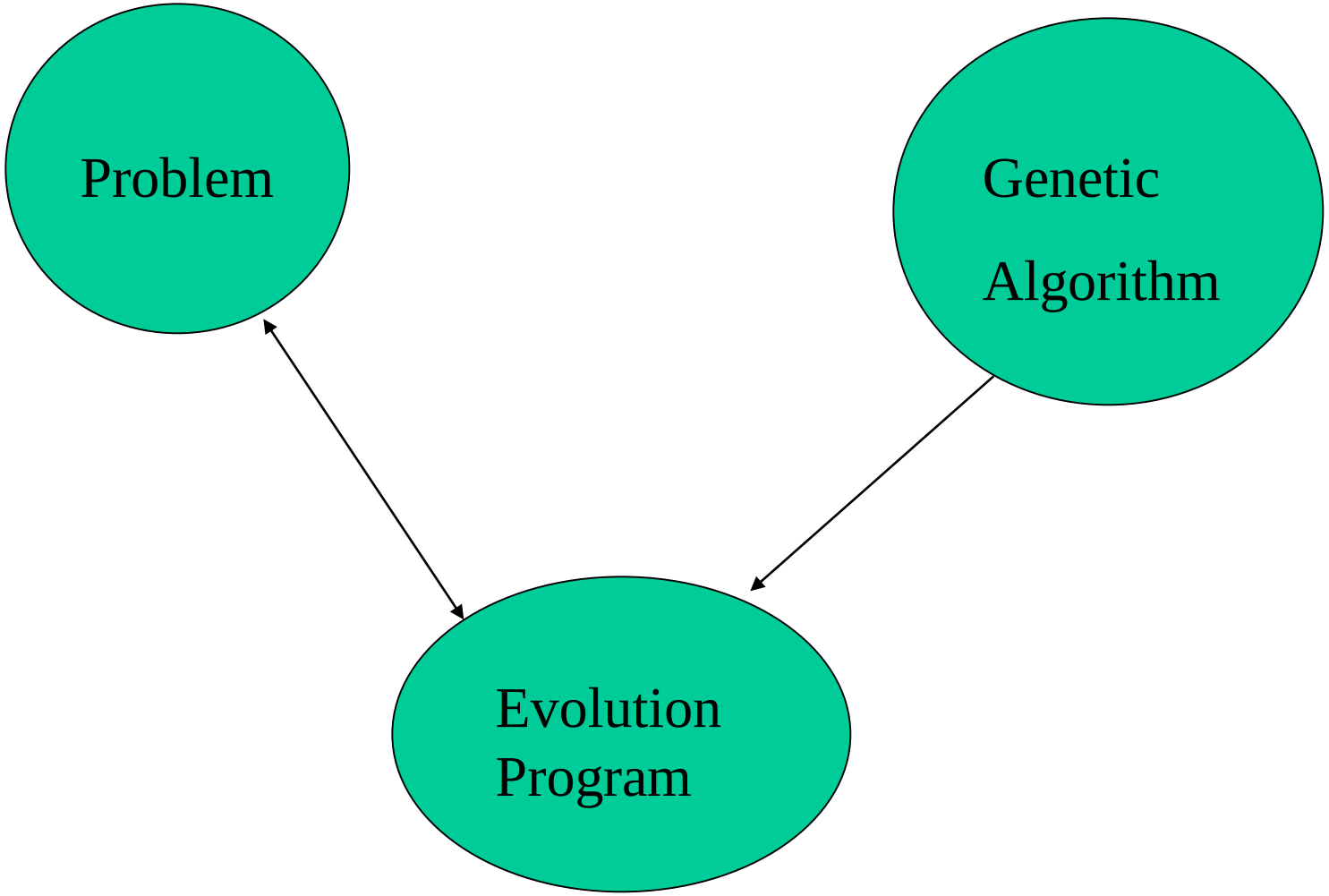
```
Procedure evolution program
begin
        t =0
        initialize P(t)
        evaluate P(t)
        while(not(termination condition) do
        begin
        t = t+1
        select P(t) from P(t-1)
        alter P(t)
        evaluate P(t)
        end
end
```

Genetic Algorithm Approach



GA operate on binary strings, require a modification of an original problem into appropriate (suitable for GA) form. This include a mapping between solution and its binary representation.

Evolution Program Approach



Evolution Programs leave the the problem unchanged, modifying a chromosome representation of a solution (using natural data structure) and applying appropriate genetic operators.

To solve a non-trivial problem using an evolution program

- transform the problem into a form appropriate for the genetic algorithm (classical GA approach)
- transform the genetic algorithm to suit the problem (second approach)

Genetic Algorithms vs. Traditional Methods.

- GAs search from a population of points, not a single point.
- GAs use payoff (objective function) information, not derivatives or other auxiliary knowledge.
- GAs use probabilistic transition rules, not deterministic rules.

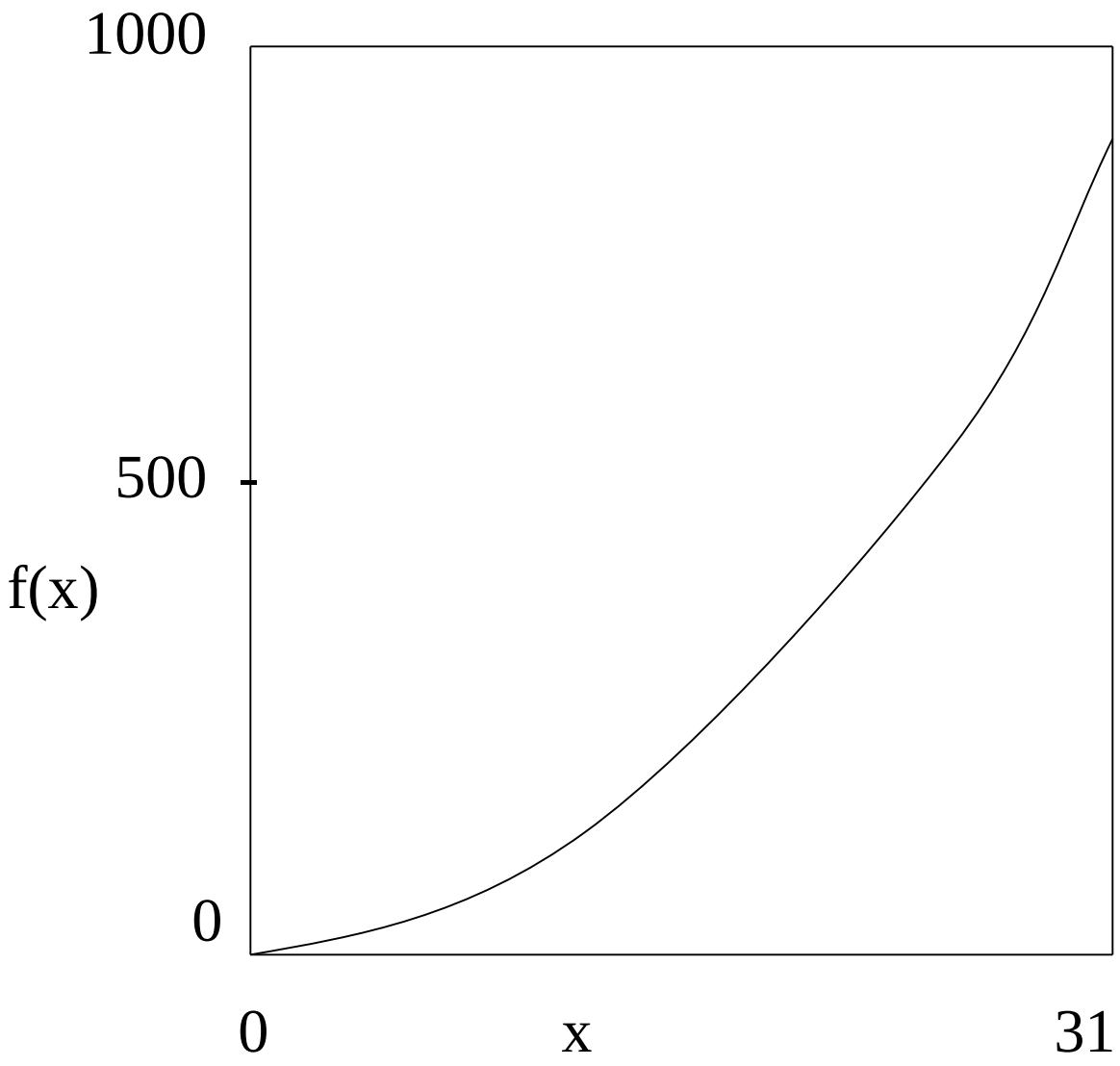
Hill climbing, simulated annealing and genetic algorithms

funny message which was presented recently on the internet

In the **Hill-climbing** method , the kangaroo can hope at best to find the top of a mountain close to where he starts. There is no guarantee that this mountain will be Everest or even a very high mountain. Various methods are used to try to find the actual.global optimum.

In **simulated annealing**, the kangaroo is drunk and hops around randomly for a long time. However, he gradually sobers up and tends to hop up hill.

In **genetic algorithms**, there are lots of kangaroos that are parachuted into the Himalayas (if the pilot did not get lost) at random places. These kangaroos do not know that they are supposed to be looking for the top of Mt. Everest. However, every few years, you shoot the kangaroos at low altitudes and hope the ones that are left will be fruitful and multiply.



f(x)= x^2 in the interval [0,31]

- As interval is of length 32, choose string length as 5($2^5 = 32$)
- Assume population size = 4
- Initial population chosen by repetitions of five coins tosses where head = 1 , Tail = 0.
- Crossover probability assumed to be unity.
- Mutation probability assumed to be 0.001.
- Expected mutations = 5*4*0.001 = 0.02

Sample problem strings and Fitness Values

No.	String	x	fitness x^2	fi/Σfi pselecti	fi*n Expected count	Actual Count
1	01101	13	169	0.14	0.58	1
2	11000	24	576	0.49	1.97	2
3	01000	8	64	0.06	0.22	0
4	10011	19	361	0.31	1.23	1
Sum			1170	1.00	4.00	4.0
Average			293	0.25	1.00	1.0
Max			576	0.49	1.97	2.0

Mating Pool	Mate	Crossover site	New Population	x	$f(x)=x^2$
01101	2	4	01100	12	144
11000	1	4	11001	25	625
11000	4	2	11011	27	729
10011	3	2	10000	16	256

1754

439

729

No.	String	x	fitness x^2	fi/Σfi pselecti	fi*n Expected count	Actual Count
1	01100	12	144	0.082	0.328	0
2	11001	25	625	0.356	1.424	1
3	11011	27	729	0.416	1.664	2
4	10000	16	256	0.146	0.584	1
Sum			1754	1.00	4.00	4.0
Average			439	0.25	1.00	1.0
Max			729	0.49	1.664	2.0

Mating Pool	Mate	Crossover site	New Population	x	$f(x)=x^2$
11001	2	3	11011	27	729
11011	1	3	11001	25	625
11011	4	2	11000	24	576
10000	3	2	10011	19	361
					2291
					573
					729

No.	String	x	fitness x^2	fi/Σfi pselecti	fi*n Expected count	Actual Count
1	11011	27	729	0.318	1.272	1
2	11001	25	625	0.273	1.092	1
3	11000	24	576	0.251	1.004	1
4	10011	19	361	0.158	0..632	1
Sum			2291	1.00	4.00	4.0
Average			573	0.25	1.00	1.0
Max			729	0.318	1.272	1.0

As new population is same as the previous population, we stop doing crossover.Select a string randomly and a bit for mutation

(Mutation of the third bit will yield a better result, closer to 31.)

Again repeat the process of crossover , finding the fitness, selecting fittest strings .

As Expected number of bit mutation per population in this example is 0.02 , in 100 generations 2 bits can be mutated.

Stopping criteria

•either fix number of generations

•or see the convergence of solution (by comparing solutions obtained by two consequent generations)

References

•Goldberg D.E. : Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, 1989

•.Michalewicz Z.: Genetic Algorithms +Data Structures =Evolution Program, Springer , 1999