CS249
WEEK-6 Slides

2201AI28	Prantik Biswas
2201AI29	Pratik Amrit
2201AI30	Raj Patil
2201AI31	Rashad Alam
2201AI32	Rishabh Verma

Random Walk

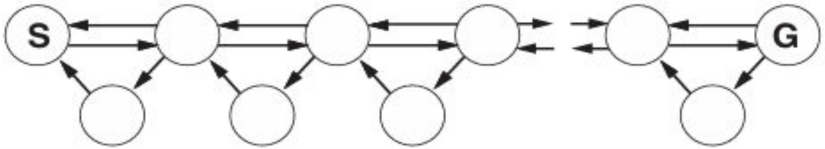


Figure 4.22 An environment in which a random walk will take exponentially many steps to find the goal.

A random walk simply selects at random one of the available actions from the current state; preference can be given to actions that have not yet been tried. It is easy to prove that a random walk will eventually find a goal or complete its exploration, provided that the space is finite.On the other hand, the process can be very slow. Figure 4.22 shows an environment in which a random walk will take exponentially many steps to find the goal because, at each step, backward progress is twice as likely as forward progress. The example is contrived, of course, but there are many real-world state spaces whose topology causes these kinds of “traps” for random walks.

Types of HC

Brief Note:

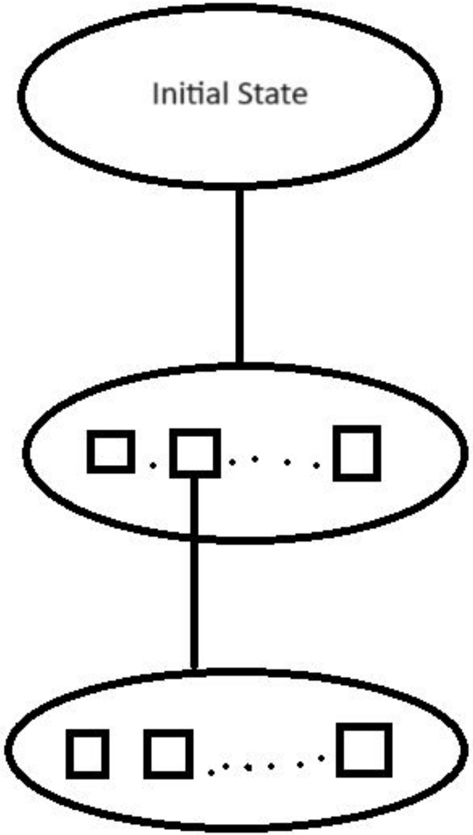
Stochastic HC >>	Stochastically Choose the best neighbour.
Random Restart >>	Run HC multiple Time.
Steepest ascent >>	In 8 Queen, Look at 56 neighbours and Choose the best one.
First Choice HC >>	First better from successors/neighbours.

Local Beam Search(LBS)

Keeping just one node in memory might seem to be an extreme reaction to the problem of memory limitations. The local beam search algorithm keeps track of k states rather than just one. It begins with k randomly generated states. At each step, all the successors of all k states are generated. If any one is a goal, the algorithm halts. Otherwise, it selects the k best successors from the complete list and repeats.

At first sight, a local beam search with k states might seem to be nothing more than running k random restarts in parallel instead of in sequence. In fact, the two algorithms are quite different. In a random-restart search, each search process runs independently of the others. In a local beam search, useful information is passed among the parallel search threads. In effect, the states that generate the best successors say to the others, “Come over here, the grass is greener!” The algorithm quickly abandons unfruitful searches and moves its resources to where the most progress is being made.

In its simplest form, local beam search can suffer from a lack of diversity among the k states—they can quickly become concentrated in a small region of the state space, making the search little more than an expensive version of hill climbing. A variant called stochastic beam search, analogous to stochastic hill climbing, helps alleviate this problem. Instead of choosing the best k from the the pool of candidate successors, stochastic beam search chooses k successors at random, with the probability of choosing a given successor being an increasing function of its value. Stochastic beam search bears some resemblance to the process of natural selection, whereby the “successors” (offspring) of a “state” (organism) populate the next generation according to its “value” (fitness).



K better neighbour state
K > 1

K better neighbour state
K = 1
LBS → Steepest Ascent HC

K = ∞
LBS → BFS

Random Restart HC(RRHC) vs LBS

	RRHC	**LBS**
Parallelism:	Sequentially	Parallely
Time Consumption:	More time consuming	Less time consuming
Starting position:	Starting position is changing	Starting position is fixed

Gradient Descent

Gradient Descent (GD) is a widely used optimization algorithm in machine learning and deep learning that minimises the cost function of a neural network model during training. It works by iteratively adjusting the weights or parameters of the model in the direction of the negative gradient of the cost function until the minimum of the cost function is reached.

We choose any starting point in weight space, and then move to a neighboring point that is downhill, repeating until we converge on the minimum possible loss.

$\mathbf{w} \leftarrow$ any point in the parameter space
loop until convergence **do**
  for each w_i **in w do**
    $$w_i \leftarrow w_i - \alpha \frac{\partial}{\partial w_i} Loss(\mathbf{w})$$

∇w

$$w_i \leftarrow w_i - \alpha \frac{\partial J(w)}{\partial w_i}$$

J(w) = cost function

Let L = |y-ŷ|

∝ = Learning rate
It is a hyperparameter

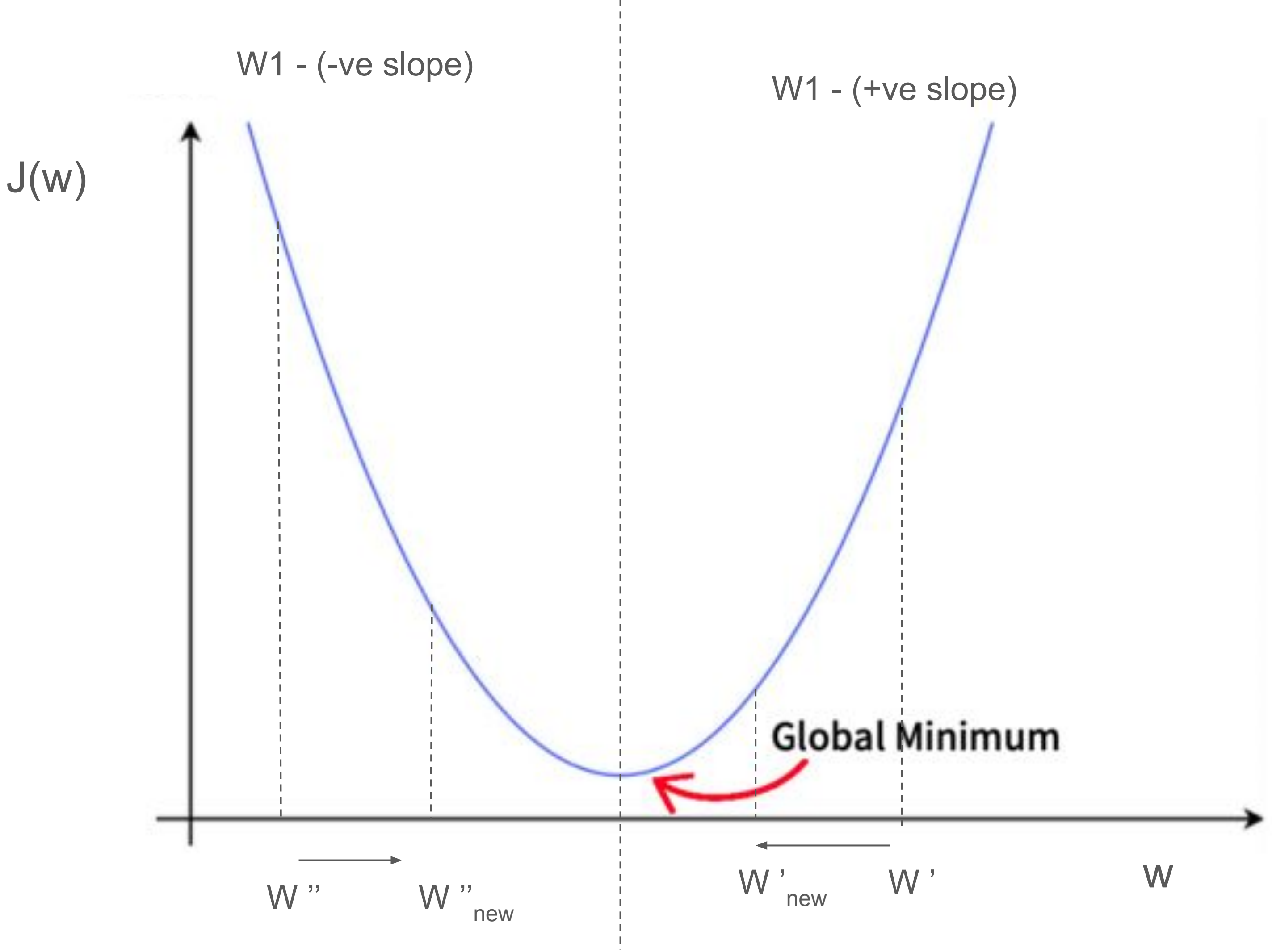
∝ ↑ may oscillate and may even diverge

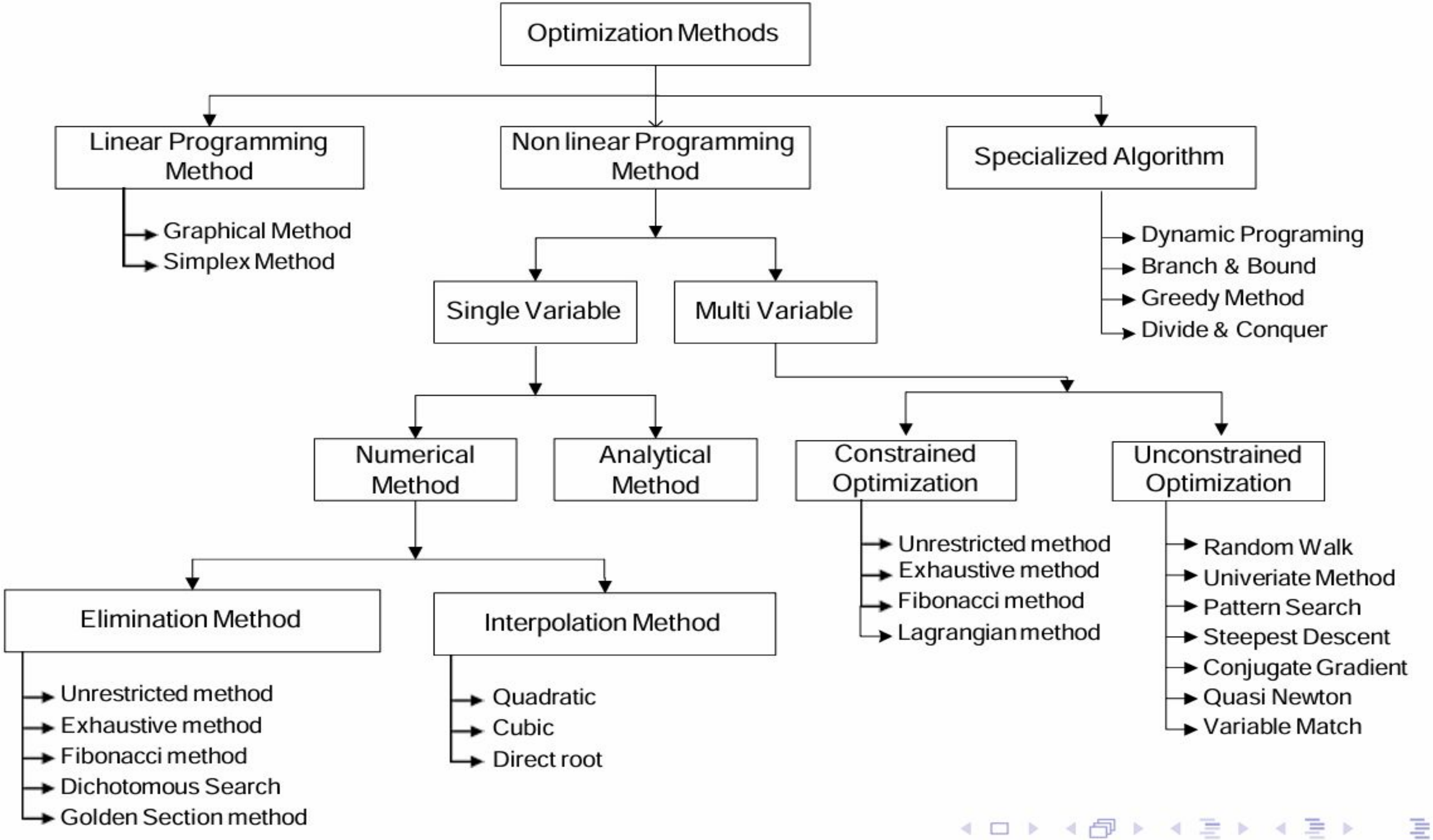
∝ ↓ may take a long time to converge

	ŷ	**y**	**L(ŷ,y)**
S1	ŷ1	y1	L1= |y1-ŷ1|
S2	ŷ2	y2	L2 = |y2-ŷ2|
…			
Sm	ŷm	ym	Lm = |ym-ŷm|

$J = \sum L_i\,(y_i,\hat{y}_i)$

Hyperparameter： In machine learning, a **hyperparameter** is a parameter, such as the learning rate or choice of optimizer, which specifies details of the learning process, hence the name *hyper*parameter It is generally set before the training begins


W1 - (-ve slope)
W1 - (+ve slope)
J(w)
Global Minimum
W ''
W ''new
W 'new
W '
w


Optimization Methods
Linear Programming Method
Non linear Programming Method
Specialized Algorithm
Graphical Method
Simplex Method
Dynamic Programing
Branch & Bound
Greedy Method
Divide & Conquer
Single Variable
Multi Variable
Numerical Method
Analytical Method
Constrained Optimization
Unconstrained Optimization
Unrestricted method
Exhaustive method
Fibonacci method
Lagrangian method
Random Walk
Univeriate Method
Pattern Search
Steepest Descent
Conjugate Gradient
Quasi Newton
Variable Match
Elimination Method
Interpolation Method
Unrestricted method
Exhaustive method
Fibonacci method
Dichotomous Search
Golden Section method
Quadratic
Cubic
Direct root

Genetic Algorithm

A genetic algorithm (or GA) is a variant of stochastic beam search in which successor states are generated by combining two parent states rather than by modifying a single state. The analogy to natural selection is the same as in stochastic beam search, except that now we are dealing with sexual rather than asexual reproduction. Like beam searches, GAs begin with a set of k randomly generated states, called the population. Each state, or individual, is represented as a string over a finite alphabet most commonly, a string of 0s and 1s. For example, an 8-queens state must specify the positions of 8 queens, each in a column of 8 squares, and so requires 8× log2 8=24 bits. Alternatively, the state could be represented as 8 digits, each in the range from 1 to 8.

GA operators

- Encoding
- Convergence
- Mating Pool
- Fitness Evaluation
- Crossover
- Mutation
- Inversion

Encoding:

- Pdf on GA encoding

For detailed info on genetic algorithms:

- Pdf on introduction to GA
- Pdf on GA operators

Evolutionary computation

In computer science, **evolutionary computation** is a family of algorithms for global optimization inspired by biological evolution, and the subfield of artificial intelligence

In evolutionary computation, an initial set of candidate solutions is generated and iteratively updated. Each new generation is produced by stochastically removing less desired solutions, and introducing small random changes as well as, depending on the method, mixing parental information. In biological terminology, a population of solutions is subjected to natural selection (or artificial selection), mutation and possibly recombination. As a result, the population will gradually evolve to increase in fitness, in this case the chosen fitness function of the algorithm.

Constraint Satisfiable Problem(CSP)

In the fields of artificial intelligence (AI) and computer science, an issue has been designated as a constraint satisfaction problem (CSP). It is described by a set of variables, a domain for each variable, and a set of constraints that outline the possible combinations of values for these variables. Finding a variable assignment that meets all of the criteria is the main objective of solving a CSP. The goal of constraint satisfaction problems is to identify values for a collection of variables that satisfy a set of limitations or guidelines.

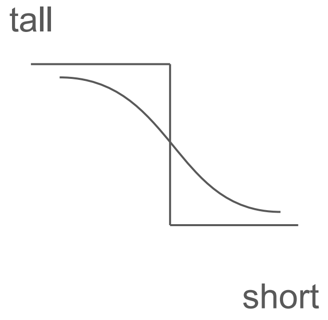
Important components in CSP are:

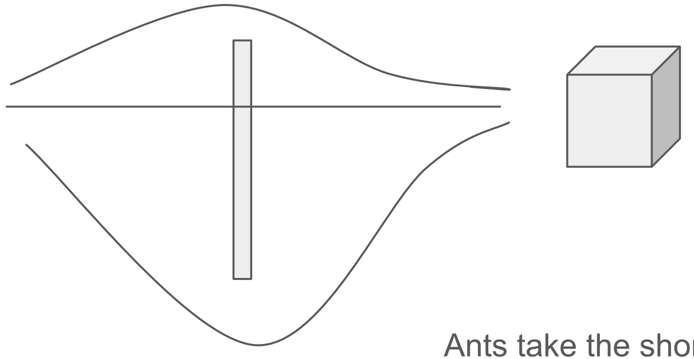
1. Variable
2. Domain
3. Constraint

1. **Loss** : It's a method of evaluating how well your algorithm models your dataset.
   Example : MAE(Mean Absolute Error),MSE(Mean Squared Error), log loss
2. **Activation** : The **activation function** of a node in an artificial neural network is a function that calculates the output of the node based on its individual inputs and their weights.
   Example: sigmoid function
3. **Optimization** : Function optimization describes a class of problems for finding the input to a given function that results in the minimum or maximum output from the function.
   Example : Gradient Descent

Fuzzy Logic

Fuzzy Logic is a form of many-valued logic in which the truth values of variables may be any real number between 0 and 1, instead of just the traditional values of true or false. It is used to deal with imprecise or uncertain information and is a mathematical method for representing vagueness and uncertainty in decision-making.





Ants take the shortest path to reach the sugar cube after straight line path is blocked

Elitist GA

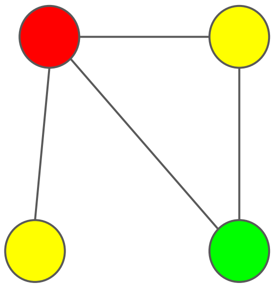
The straightforward inclusion of the best individuals of a generation in the population of the following generation

Thus, before applying elitism to create new generations of individuals, we must evaluate the current generation to find the best individuals. For that, evolutionary algorithms calculate and compare an optimization function (or objective function) for each available individual of a population.

Graph Coloring Problem

Graph coloring refers to the problem of coloring vertices of a graph in such a way that no two adjacent vertices have the same color. This is also called the vertex coloring problem. If coloring is done using at most m colors, it is called m-coloring.

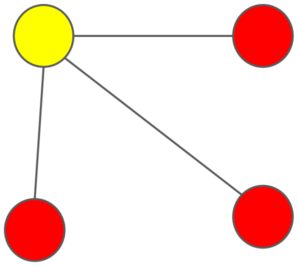
Our objective is to minimize k



GA encoding

c1	c2	c3	c2
v1	v2	v3	v4

K = 3



K = 2

GA encoding

c1	c2	c2	c2
v1	v2	v3	v4

Encoding for TSP

Suppose there are 5 cities: 0, 1, 2, 3, 4. The salesman is in city 0 and he has to find the shortest route to travel through all the cities back to the city 0. A chromosome representing the path chosen can be represented as

Initial solution

0	1	4	3	2	0

Mutation

0	1	4	2	3	0

→

0	1	3	2	4	0

References

- Google:www.google.com
- GeeksForGeeks : www.geeksforgeeks.com
- Artificial Intelligence - A Modern Approach by Stuart Russell and Peter Norvig
- Slides : From professor Chandranath Adak

Contributions

2201AI28 Prantik Biswas	20%
2201AI29 Pratik Amrit	20%
2201AI30 Raj Patil	20%
2201AI31 Rashad Alam	20%
2201AI32 Rishabh Verma	20%