IS-ZC444: ARTIFICIAL INTELLIGENCE

Lecture-06: Beyond Classical Search

Dr. Kamlesh Tiwari
Assistant Professor
Department of Computer Science and Information Systems,
BITS Pilani, Pilani, Jhunjhunu-333031, Rajasthan, INDIA

September 12, 2018 (WILP @ BITS-Pilani Jul-Nov 2018)

Beyond Classical Search

We have seen **systematic search** algorithms for problems that are observable and deterministic. That may not be for real-world problem.

Beyond Classical Search

We have seen **systematic search** algorithms for problems that are observable and deterministic. That may not be for real-world problem.

- In many problems, path to the goal is irrelevant.
  - 8-queens - order of adding queens is not important
  - Integrated circuit design
  - Factory floor layout
  - Job shop scheduling
  - Automatic programming
  - Telecommunication network optimization
  - Vehicle Routing
  - Portfolio management

Beyond Classical Search

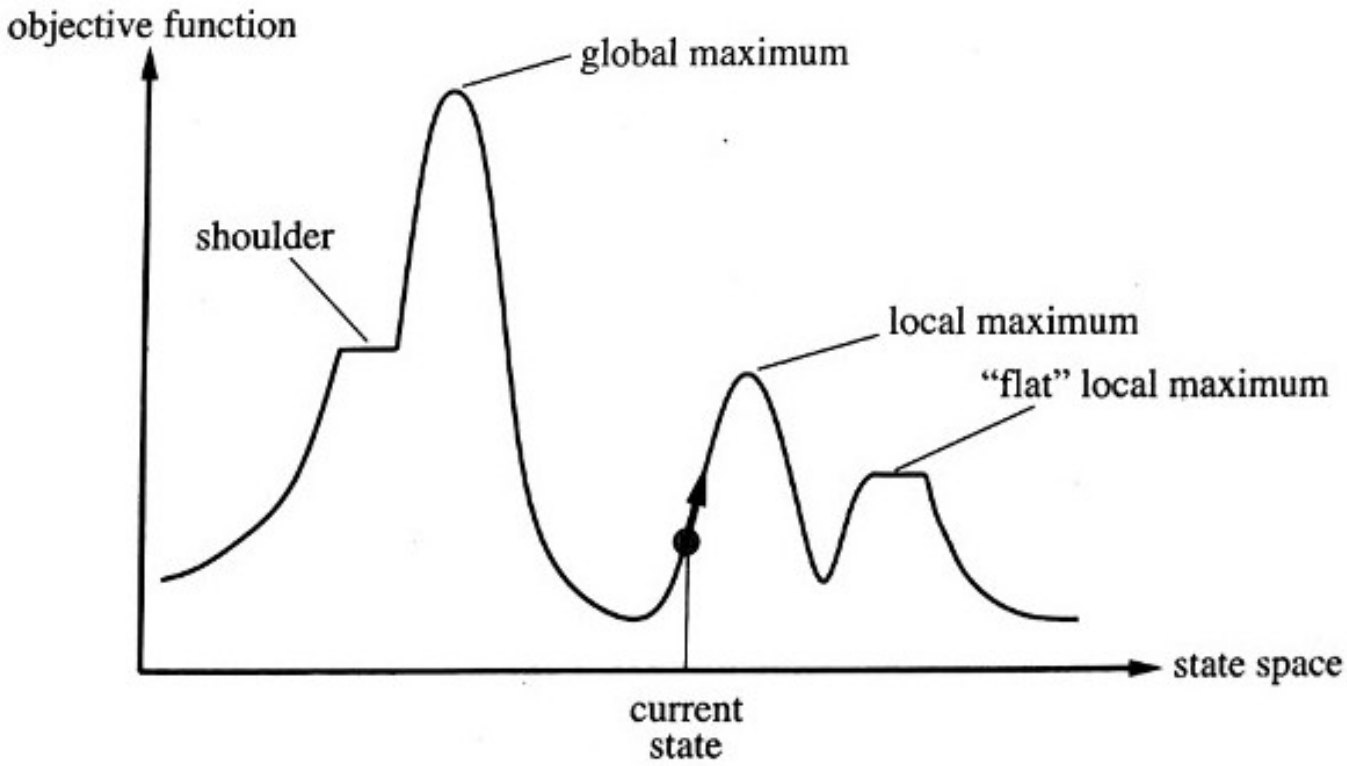
We have seen **systematic search** algorithms for problems that are observable and deterministic. That may not be for real-world problem.

- In many problems, path to the goal is irrelevant.
  - 8-queens - order of adding queens is not important
  - Integrated circuit design
  - Factory floor layout
  - Job shop scheduling
  - Automatic programming
  - Telecommunication network optimization
  - Vehicle Routing
  - Portfolio management
- We go for **local search** algorithms in such scenarios
- These algorithms operate using a single current node (and move to its neighbor). Advantages include
  1. Uses very little memory
  2. Can find reasonable solution in large or infinite (continuous) state space.

Local Search Algorithms

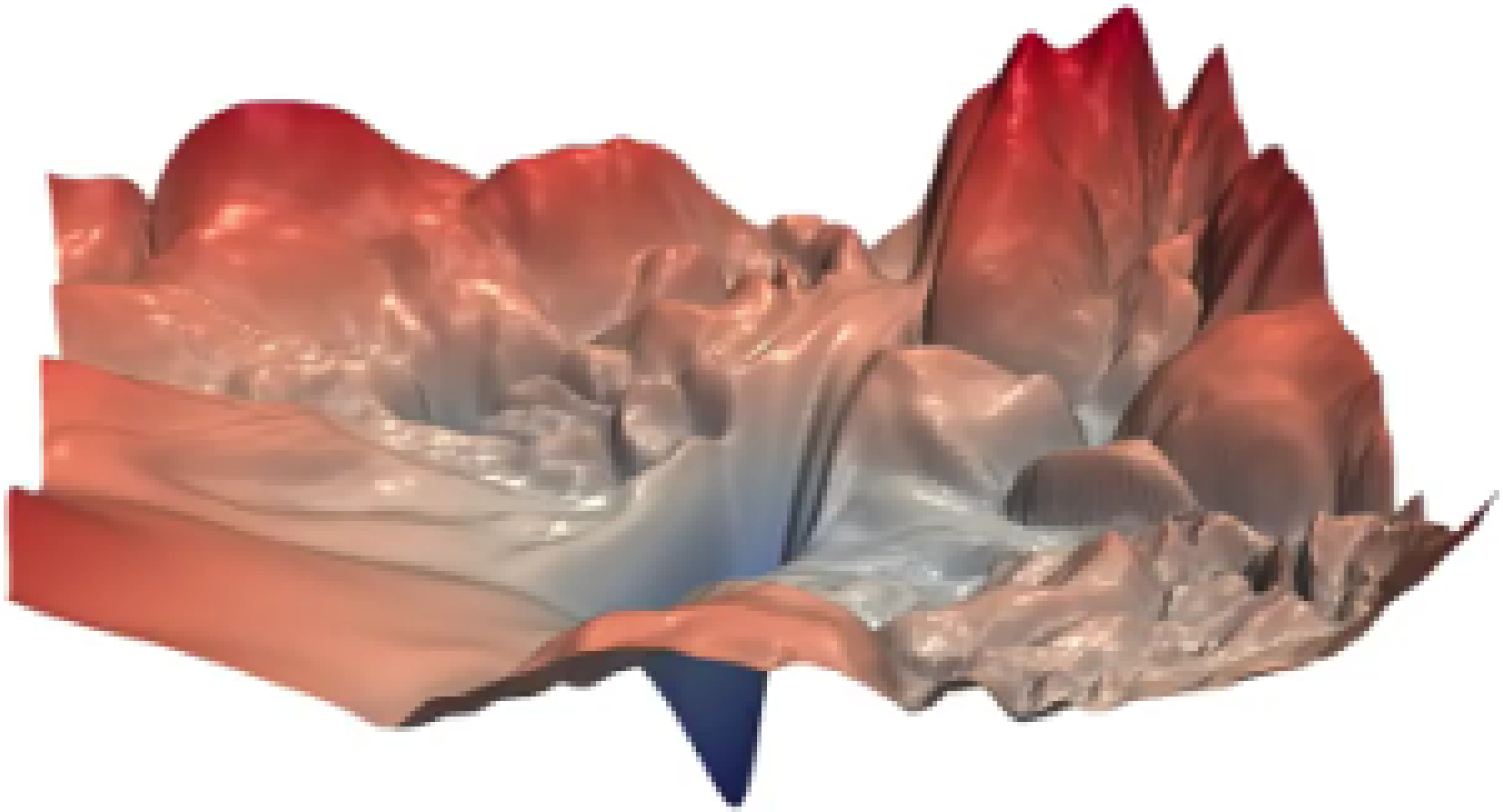
- Local search algorithms can solve pure optimization problems.
- They find optimum state based on some **objective function** [1]

State space landscape can be used to proceed.



[1] Many optimization problems do not fit in standard search model. For example Darwinian evolution uses an objective function **reproductive fitness** but there is no goal test and path cost for this.

Real State Space Landscape



Hill Climbing Search

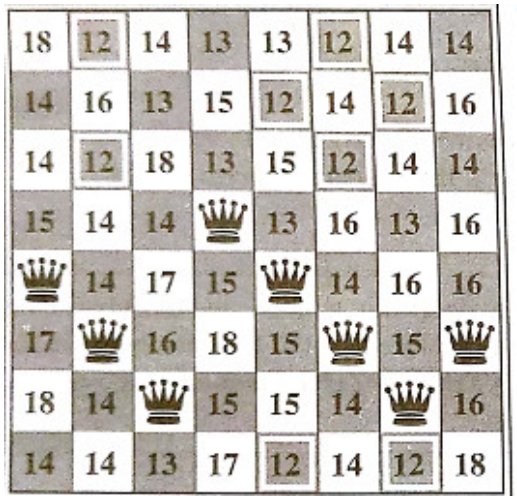
- Steepest ascent version
- Look around and choose the best
- Greedy-local search
- Makes quite rapid progress

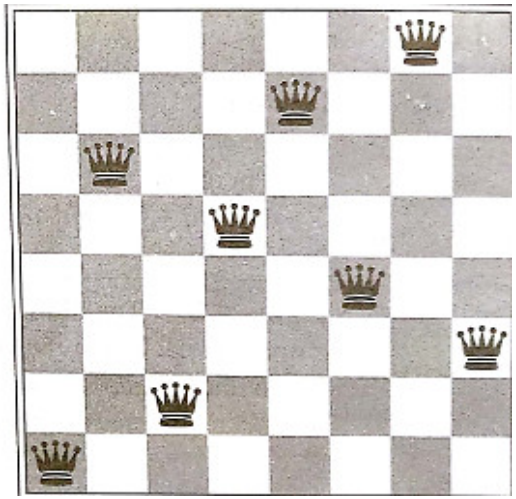
Algorithm 1: Hill-Climbing(problem)

```
1  Returned a state of problem that is local maximum.
2   current ← Make-Node (problem.Initial-State)
3  while not terminate do
4      neighbor ← a highest valued successor of current
5      if neighbor.Value ≤ current.Value then
6          return current.State
7      current ← neighbor
```

Hill Climbing Search

- Consider 8-queens problem
- In **complete-state formulation** has 8-queens in each state (one per column)
- Successor of a state are all possible states generated by moving a single queen to another square (each queen have 7 moves) so $8 \times 7 = 56$ successors.
- Heuristic cost function h is the number of pairs of queens that are attacking each other ($h = 17$ to $h = 1$)





Gets Stuck

Sometime algorithm reaches to a state where no further progress could be made.

Due to

1. Local Maxima
2. Ridges
3. Plateaux

Gets Stuck

Sometime algorithm reaches to a state where no further progress could be made.

Due to

1. Local Maxima
2. Ridges
3. Plateaux

Starting from randomly generated 8-queens state, hill climbing gets stuck 86% of time. Taking 4-steps when succeed and 3-steps when stuck (8^8 states in state space)

Gets Stuck

Sometime algorithm reaches to a state where no further progress could be made.

Due to

1. Local Maxima
2. Ridges
3. Plateaux

Starting from randomly generated 8-queens state, hill climbing gets stuck 86% of time. Taking 4-steps when succeed and 3-steps when stuck (8^8 states in state space)

Allow **sideway moves** in this scenario

Gets Stuck

Sometime algorithm reaches to a state where no further progress could be made.

Due to

1. Local Maxima
2. Ridges
3. Plateaux

Starting from randomly generated 8-queens state, hill climbing gets stuck 86% of time. Taking 4-steps when succeed and 3-steps when stuck (8^8 states in state space)

Allow **sideway moves** in this scenario

How many sideway moves? ∞

Gets Stuck

Sometime algorithm reaches to a state where no further progress could be made.

Due to

1. Local Maxima
2. Ridges
3. Plateaux

Starting from randomly generated 8-queens state, hill climbing gets stuck 86% of time. Taking 4-steps when succeed and 3-steps when stuck (8^8 states in state space)

Allow **sideway moves** in this scenario

How many sideway moves? ∞
Put a limit on this (say 100) Now it solves 94% problems. But, takes 21 steps to solve and 64 for failure on an average.

Hill Climbing Variants

Stochastic hill climbing

Instead of the best one, choose a path with probability of its steepness (converge slowly).

Hill Climbing Variants

Stochastic hill climbing

Instead of the best one, choose a path with probability of its steepness (converge slowly).

First-choice hill climbing

Do not explore the state-space when you see the first improvement.

Hill Climbing Variants

Stochastic hill climbing

Instead of the best one, choose a path with probability of its steepness (converge slowly).

First-choice hill climbing

Do not explore the state-space when you see the first improvement.

Random-restart hill climbing

if probability of success is p then run $1/p$ number of times.

- For our previous case [a] it needs 7 iterations.
- and 22 steps

[a] With random start 8-queens state, hill climbing gets stuck 86% of time.

Simulated Annealing (Metallurgy approach)

- Simulated annealing is hill climbing combined with **random walk**
- Step size is gradually reduced
- First Applied around 1980, for VLSI layout problem

Algorithm 2: Simulated-Annealing(problem, schedule)

```
1 current ← Make-Node (problem.Initial-State)
2 for t = 1 to ∞ do
3     T ← schedule(t)
4     if T = 0 then return current
5     next ← a randomly selected successor of current
6     ΔE ← next.Value - current.Value
7     if ΔE > 0 then current ← next
8     else current ← next with probability e^(ΔE/T)
```

Genetic Algorithms

Evolution

- Recall Darwin's theory of evolution: "*Survival of the fittest*" [2]

[2] Images taken from various sources on Internet

Genetic Algorithms

Evolution

- Recall Darwin's theory of evolution: "*Survival of the fittest*" [2]





[2] Images taken from various sources on Internet

Genetic Algorithms

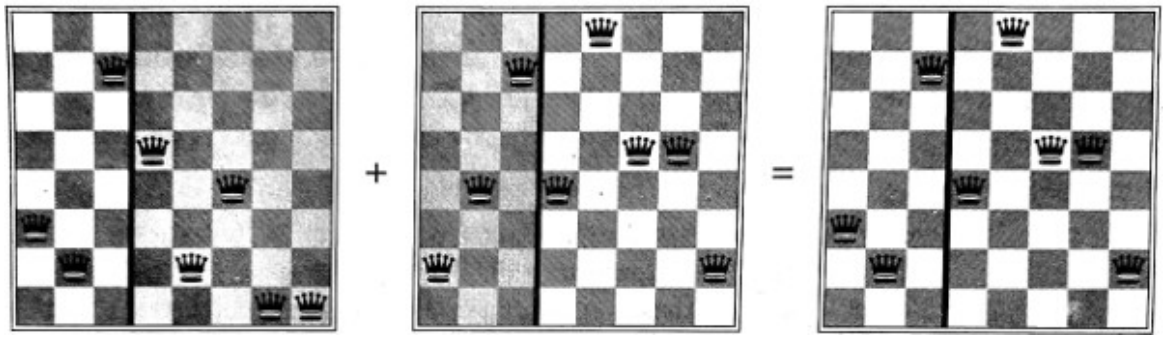
Evolution

- Recall Darwin's theory of evolution: "*Survival of the fittest*" [2]





Is it true? Let's formulate and try..



[2] Images taken from various sources on Internet

Genetic Algorithms

Learning approach of Genetic algorithms is based on simulated evolution (appeared in 1975)

Genetic Algorithms

Learning approach of Genetic algorithms is based on simulated evolution (appeared in 1975)

- State are represented using fixed length bit strings (chromosome)

Genetic Algorithms

Learning approach of Genetic algorithms is based on simulated evolution (appeared in 1975)

- State are represented using fixed length bit strings (chromosome)
- Search for a goal state begins with a population of initial states

Genetic Algorithms

Learning approach of Genetic algorithms is based on simulated evolution (appeared in 1975)

- State are represented using fixed length bit strings (chromosome)
- Search for a goal state begins with a population of initial states
  - Members of the current population give rise to the next generation population using random **mutation** and **crossover**

Genetic Algorithms

Learning approach of Genetic algorithms is based on simulated evolution (appeared in 1975)

- State are represented using fixed length bit strings (chromosome)
- Search for a goal state begins with a population of initial states
  - Members of the current population give rise to the next generation population using random **mutation** and **crossover**
  - States are evaluated using some **fitness** measure

Genetic Algorithms

Learning approach of Genetic algorithms is based on simulated evolution (appeared in 1975)

- State are represented using fixed length bit strings (chromosome)
- Search for a goal state begins with a population of initial states
  - Members of the current population give rise to the next generation population using random **mutation** and **crossover**
  - States are evaluated using some **fitness** measure
  - Most fit state act as a seeds for producing next generation

Genetic Algorithms

Learning approach of Genetic algorithms is based on simulated evolution (appeared in 1975)

- State are represented using fixed length bit strings (chromosome)
- Search for a goal state begins with a population of initial states
  - Members of the current population give rise to the next generation population using random **mutation** and **crossover**
  - States are evaluated using some **fitness** measure
  - Most fit state act as a seeds for producing next generation
- Applied a variety of learning tasks and optimization problems (like robot control and learning parameters for ANN)
- Search can move abruptly. Crowding can happen

Genetic Algorithms

Learning approach of Genetic algorithms is based on simulated evolution (appeared in 1975)

- State are represented using fixed length bit strings (chromosome)
- Search for a goal state begins with a population of initial states
  - Members of the current population give rise to the next generation population using random **mutation** and **crossover**
  - States are evaluated using some **fitness** measure
  - Most fit state act as a seeds for producing next generation
- Applied a variety of learning tasks and optimization problems (like robot control and learning parameters for ANN)
- Search can move abruptly. Crowding can happen

Without **guarantee**, GA often finds an object of high fitness

Genetic Algorithms

Algorithm 3: GA (`Fitness`, F_{th}, p, r, m)

1 P $\leftarrow$ generate p states at random
2 **while** $max(\texttt{Fitness}(h_1), \texttt{Fitness}(h_2), ..., \texttt{Fitness}(h_p)) < F_{th}$ **do**
3 **Select:** $(1 - r)p$ members of P
4 **Crossover:** on $(r \times p)/2$ pairs to produce two offspring
5 **Mutation:** randomly invert a bit of m percentage of population
6 **return** state h_i having maximum `Fitness`(h_i)

Genetic Algorithms

Algorithm 4: GA (`Fitness`, F_{th}, p, r, m)

1 P $\leftarrow$ generate p states at random
2 **while** $max(\texttt{Fitness}(h_1), \texttt{Fitness}(h_2), ..., \texttt{Fitness}(h_p)) < F_{th}$ **do**
3 **Select:** $(1 - r)p$ members of P
4 **Crossover:** on $(r \times p)/2$ pairs to produce two offspring
5 **Mutation:** randomly invert a bit of m percentage of population
6 **return** state h_i having maximum `Fitness`(h_i)

- Fitness function is typically a heuristic
- The fitness function is a criterion for ranking states to select states probabilistically for inclusion in the next generation population

Selection

- **Fitness proportionate**. Probability of selecting a state in next generation is

$$Pr(h_i) = \frac{\texttt{Fitness}(h_i)}{\sum_{j=1}^{p} \texttt{Fitness}(h_j)}$$

Selection

- **Fitness proportionate**. Probability of selecting a state in next generation is

$$Pr(h_i) = \frac{\texttt{Fitness}(h_i)}{\sum_{j=1}^{p} \texttt{Fitness}(h_j)}$$

- **Tournament selection**. randomly pick two states and then with some predefined probability p the more fit of these two is then selected, and with probability $(1 - p)$ the less fit state is selected

Selection

- **Fitness proportionate**. Probability of selecting a state in next generation is

$$Pr(h_i) = \frac{\texttt{Fitness}(h_i)}{\sum_{j=1}^{p} \texttt{Fitness}(h_j)}$$

- **Tournament selection**. randomly pick two states and then with some predefined probability p the more fit of these two is then selected, and with probability $(1 - p)$ the less fit state is selected
- **Rank selection**. states are sorted by fitness and the probability of selection of a state is proportional to its rank in this sorted list, rather than its fitness

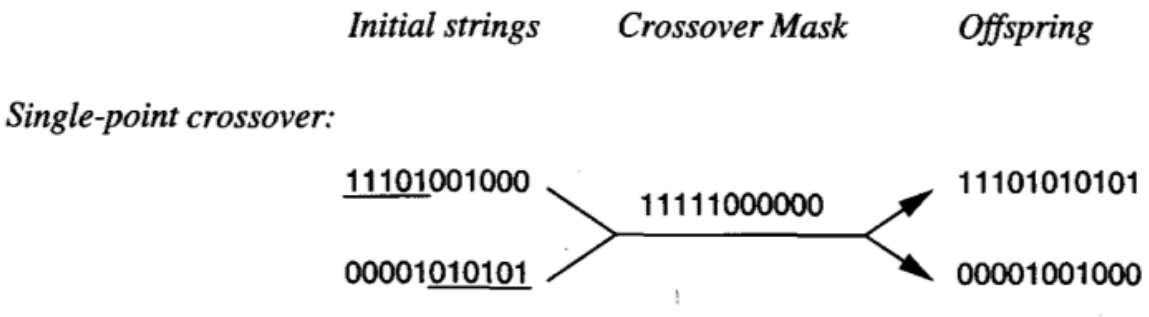
Selection

- **Fitness proportionate**. Probability of selecting a state in next generation is

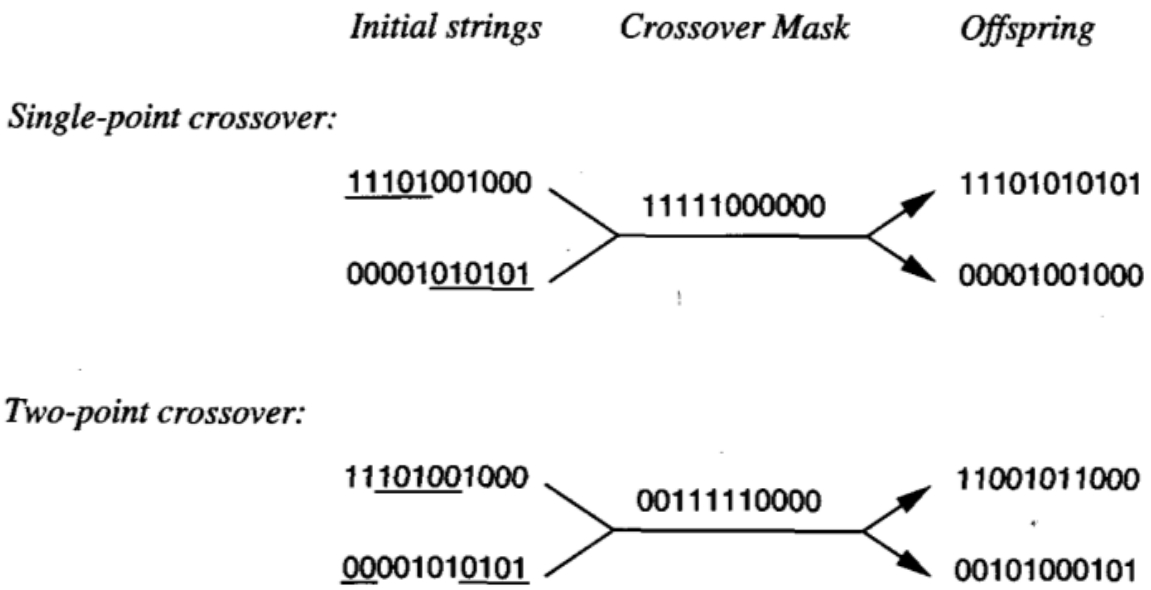
$$Pr(h_i) = \frac{\texttt{Fitness}(h_i)}{\sum_{j=1}^{p} \texttt{Fitness}(h_j)}$$

- **Tournament selection**. randomly pick two states and then with some predefined probability p the more fit of these two is then selected, and with probability $(1 - p)$ the less fit state is selected
- **Rank selection**. states are sorted by fitness and the probability of selection of a state is proportional to its rank in this sorted list, rather than its fitness
- **Elitist Model**. select a small proportion of the fittest candidates in current population intact into the next generation

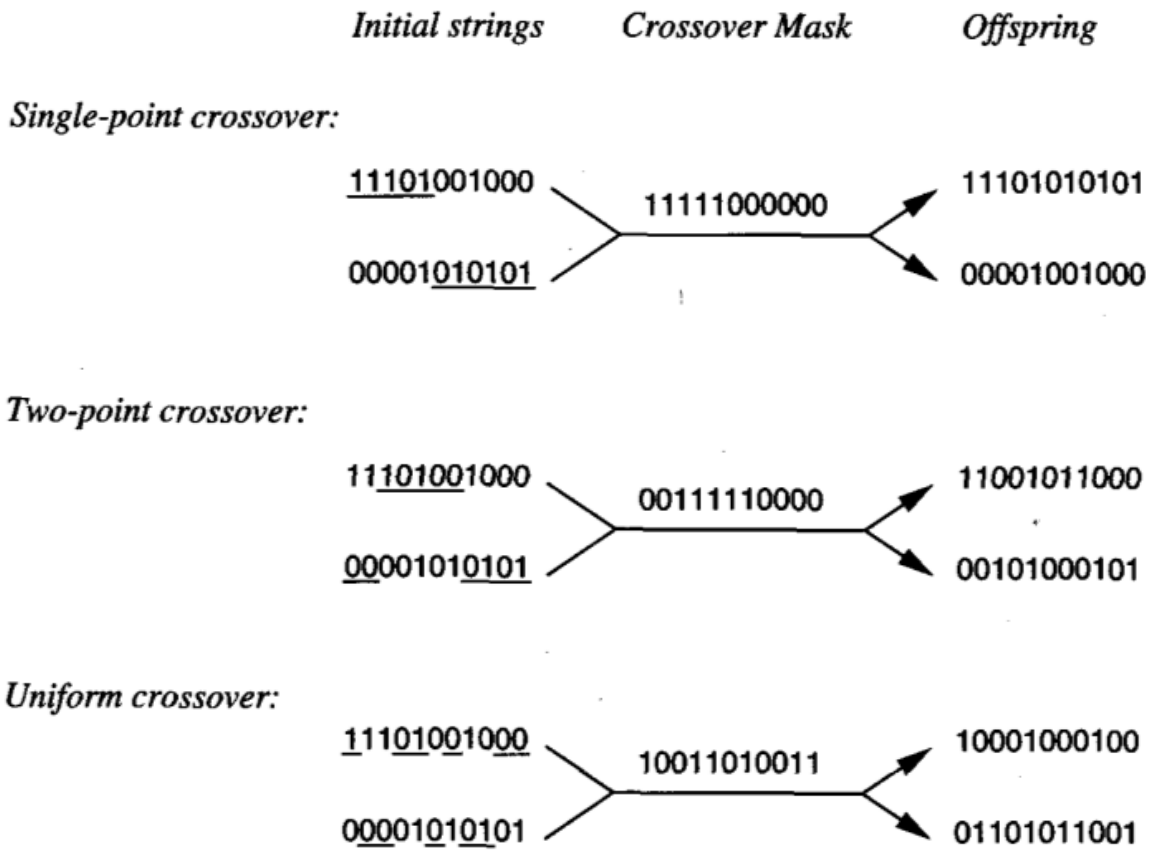
Genetic Operators (crossover and mutation)



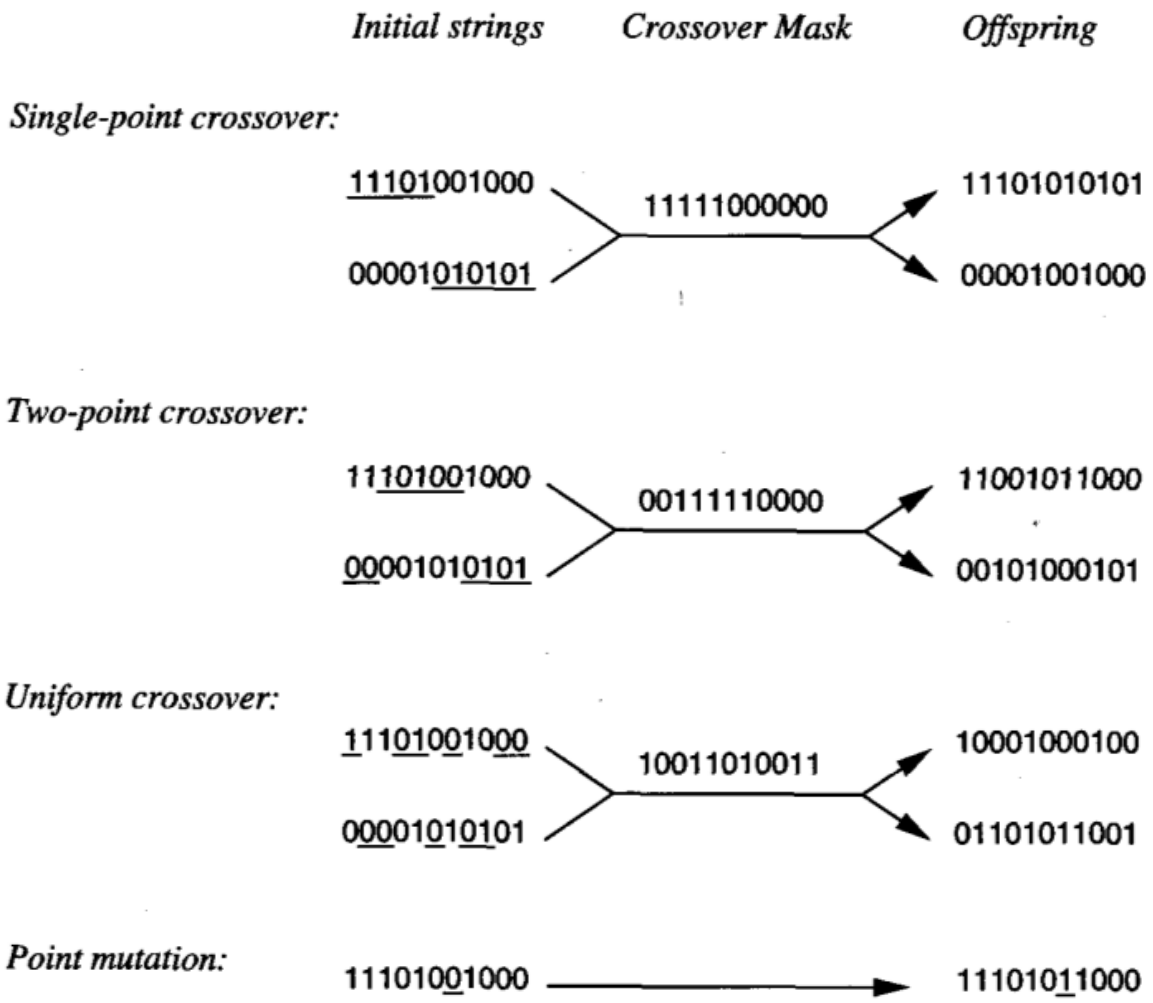
Genetic Operators (crossover and mutation)



Genetic Operators (crossover and mutation)



Genetic Operators (crossover and mutation)



Does GA works?

- Can we mathematically characterize the evolution

Does GA works?

- Can we mathematically characterize the evolution
- **Schema:** string of 0, 1 or $*$

Does GA works?

- Can we mathematically characterize the evolution
- **Schema:** string of 0, 1 or $*$ like $0*1$

Does GA works?

- Can we mathematically characterize the evolution
- **Schema:** string of 0, 1 or $*$ like $0*1$ denoting set $\{001, 011\}$
- String 1011 is **representative** of 2^4 schema

Does GA works?

- Can we mathematically characterize the evolution
- **Schema:** string of 0, 1 or $*$ like $0*1$ denoting set $\{001, 011\}$
- String 1011 is **representative** of 2^4 schema
- Let $m(s, t)$ be number of instances of schema s in generation t

Does GA works?

- Can we mathematically characterize the evolution
- **Schema:** string of 0, 1 or $*$ like $0*1$ denoting set $\{001, 011\}$
- String 1011 is **representative** of 2^4 schema
- Let $m(s, t)$ be number of instances of schema s in generation t
- Consider *selection*, let fitness of individual h be $f(h)$

Does GA works?

- Can we mathematically characterize the evolution
- **Schema:** string of 0, 1 or $*$ like $0*1$ denoting set $\{001, 011\}$
- String 1011 is **representative** of 2^4 schema
- Let $m(s, t)$ be number of instances of schema s in generation t
- Consider *selection*, let fitness of individual h be $f(h)$ and average fitness of whole n size population

Does GA works?

- Can we mathematically characterize the evolution
- **Schema:** string of 0, 1 or $*$ like $0*1$ denoting set $\{001, 011\}$
- String 1011 is **representative** of 2^4 schema
- Let $m(s, t)$ be number of instances of schema s in generation t
- Consider *selection*, let fitness of individual h be $f(h)$ and average fitness of whole n size population at time t

Does GA works?

- Can we mathematically characterize the evolution
- **Schema:** string of 0, 1 or $*$ like $0*1$ denoting set $\{001, 011\}$
- String 1011 is **representative** of 2^4 schema
- Let $m(s, t)$ be number of instances of schema s in generation t
- Consider *selection*, let fitness of individual h be $f(h)$ and average fitness of whole n size population at time t be $\bar{f}(t)$

Does GA works?

- Can we mathematically characterize the evolution
- **Schema:** string of 0, 1 or $*$ like $0*1$ denoting set $\{001, 011\}$
- String 1011 is **representative** of 2^4 schema
- Let $m(s, t)$ be number of instances of schema s in generation t
- Consider *selection*, let fitness of individual h be $f(h)$ and average fitness of whole n size population at time t be $\bar{f}(t)$
- $h \in s \cup p_t$ means 1) h is representative of s and 2) it is present in the population at time t

Does GA works?

- Can we mathematically characterize the evolution
- **Schema:** string of 0, 1 or $*$ like $0*1$ denoting set $\{001, 011\}$
- String 1011 is **representative** of 2^4 schema
- Let $m(s,t)$ be number of instances of schema s in generation t
- Consider *selection*, let fitness of individual h be $f(h)$ and average fitness of whole n size population at time t be $\bar{f}(t)$
- $h \in s \cup p_t$ means 1) h is representative of s and 2) it is present in the population at time t
- Let $\hat{u}(s,t)$ be average fitness of instances of s at time t

Does GA works?

- Can we mathematically characterize the evolution
- **Schema:** string of 0, 1 or $*$ like $0*1$ denoting set $\{001, 011\}$
- String 1011 is **representative** of 2^4 schema
- Let $m(s,t)$ be number of instances of schema s in generation t
- Consider *selection*, let fitness of individual h be $f(h)$ and average fitness of whole n size population at time t be $\bar{f}(t)$
- $h \in s \cup p_t$ means 1) h is representative of s and 2) it is present in the population at time t
- Let $\hat{u}(s,t)$ be average fitness of instances of s at time t

$$\hat{u}(s,t) = \frac{\sum_{h \in s \cup p_t} f(h)}{m(s,t)}$$

Does GA works?

- Can we mathematically characterize the evolution
- **Schema:** string of 0, 1 or $*$ like $0*1$ denoting set $\{001, 011\}$
- String 1011 is **representative** of 2^4 schema
- Let $m(s,t)$ be number of instances of schema s in generation t
- Consider *selection*, let fitness of individual h be $f(h)$ and average fitness of whole n size population at time t be $\bar{f}(t)$
- $h \in s \cup p_t$ means 1) h is representative of s and 2) it is present in the population at time t
- Let $\hat{u}(s,t)$ be average fitness of instances of s at time t

$$\hat{u}(s,t) = \frac{\sum_{h \in s \cup p_t} f(h)}{m(s,t)}$$

- We know $Pr(h) = f(h)/(\sum f(h)) = f(h)/(n\bar{f}(t))$

Does GA works? (contd..)

- Probability that we will select a representative of schema s is

$$Pr(h \in s) = \sum_{h \in s \cup p_t} Pr(h)$$

Does GA works? (contd..)

- Probability that we will select a representative of schema s is

$$Pr(h \in s) = \sum_{h \in s \cup p_t} Pr(h) = \sum_{h \in s \cup p_t} f(h)/(n\bar{f}(t))$$

Does GA works? (contd..)

- Probability that we will select a representative of schema s is

$$\begin{aligned} Pr(h \in s) &= \sum_{h \in s \cup p_t} Pr(h) = \sum_{h \in s \cup p_t} f(h)/(n\bar{f}(t)) \\ &= \frac{\hat{u}(s,t)}{n\bar{f}(t)} m(s,t) \end{aligned}$$

Does GA works? (contd..)

- Probability that we will select a representative of schema s is

$$\begin{aligned} Pr(h \in s) &= \sum_{h \in s \cup p_t} Pr(h) = \sum_{h \in s \cup p_t} f(h)/(n\bar{f}(t)) \\ &= \frac{\hat{u}(s,t)}{n\bar{f}(t)} m(s,t) \end{aligned}$$

- Expected number of instances of s resulting from the n independent selection steps that create the entire new generation is just n times this probability.

Does GA works? (contd..)

- Probability that we will select a representative of schema s is

$$\begin{aligned} Pr(h \in s) &= \sum_{h \in s \cup p_t} Pr(h) = \sum_{h \in s \cup p_t} f(h)/(n\bar{f}(t)) \\ &= \frac{\hat{u}(s,t)}{n\bar{f}(t)} m(s,t) \end{aligned}$$

- Expected number of instances of s resulting from the n independent selection steps that create the entire new generation is just n times this probability. Therefore,

$$E[m(s,t+1)] = \frac{\hat{u}(s,t)}{\bar{f}(t)} m(s,t)$$

Does GA works? (contd..)

$$E[m(s, t+1)] = \frac{\hat{u}(s,t)}{\bar{f}(t)} m(s,t)$$

Expected number of representative instances of a schema s in the generation at time $t+1$ is

Does GA works? (contd..)

$$E[m(s, t+1)] = \frac{\hat{u}(s,t)}{\bar{f}(t)} m(s,t)$$

Expected number of representative instances of a schema s in the generation at time $t+1$ is

1. Proportional to the average fitness $\hat{u}(s,t)$ of instances of this schema at time t

Does GA works? (contd..)

$$E[m(s, t+1)] = \frac{\hat{u}(s,t)}{\bar{f}(t)} m(s,t)$$

Expected number of representative instances of a schema s in the generation at time $t+1$ is

1. Proportional to the average fitness $\hat{u}(s,t)$ of instances of this schema at time t , and
2. Inversely proportional to the average fitness $\bar{f}(t)$ of all members of the population at time t

Does GA works? (contd..)

$$E[m(s, t+1)] = \frac{\hat{u}(s,t)}{\bar{f}(t)} m(s,t)$$

Expected number of representative instances of a schema s in the generation at time $t+1$ is

1. Proportional to the average fitness $\hat{u}(s,t)$ of instances of this schema at time t , and
2. Inversely proportional to the average fitness $\bar{f}(t)$ of all members of the population at time t

Thus, we can expect schema with above average fitness to be represented with increasing frequency on successive generations

Does GA works? (contd..)

Also consider negative effects of single point crossover and mutation

Does GA works? (contd..)

Also consider negative effects of single point crossover and mutation

- Let p_c represents the probability of **crossover** on an individual. $d(s)$ be the distance between left most and right most defined bit of s and l be the length of individual bit string

Does GA works? (contd..)

Also consider negative effects of single point crossover and mutation

- Let p_c represents the probability of **crossover** on an individual. $d(s)$ be the distance between left most and right most defined bit of s and l be the length of individual bit string
- Let p_m represents the probability of **mutation** on an individual and $o(s)$ be number of defined bits in s

Does GA works? (contd..)

Also consider negative effects of single point crossover and mutation

- Let p_c represents the probability of **crossover** on an individual. $d(s)$ be the distance between left most and right most defined bit of s and l be the length of individual bit string
- Let p_m represents the probability of **mutation** on an individual and $o(s)$ be number of defined bits in s

Full schema theorem thus provides a lower bound on the expected frequency of schema s, as follows

$$E[m(s,t+1)] \geq \frac{\hat{u}(s,t)}{\bar{f}(t)} m(s,t)\left(1 - p_c \frac{d(s)}{l-1}\right)(1-p_m)^{o(s)}$$

Does GA works? (contd..)

Also consider negative effects of single point crossover and mutation

- Let p_c represents the probability of **crossover** on an individual. $d(s)$ be the distance between left most and right most defined bit of s and l be the length of individual bit string
- Let p_m represents the probability of **mutation** on an individual and $o(s)$ be number of defined bits in s

Full schema theorem thus provides a lower bound on the expected frequency of schema s, as follows

$$E[m(s,t+1)] \geq \frac{\hat{u}(s,t)}{\bar{f}(t)} m(s,t)\left(1 - p_c \frac{d(s)}{l-1}\right)(1-p_m)^{o(s)}$$

Similar expression. More fit schemas will tend to grow in influence.

Thank You!

Thank you very much for your attention!

Queries ?

(Reference[3])

[3] 1) Book - *AIMA*, ch-04, Russell and Norvig. 2) Book - Machine Learning, ch-09, *Tom Mitchell*