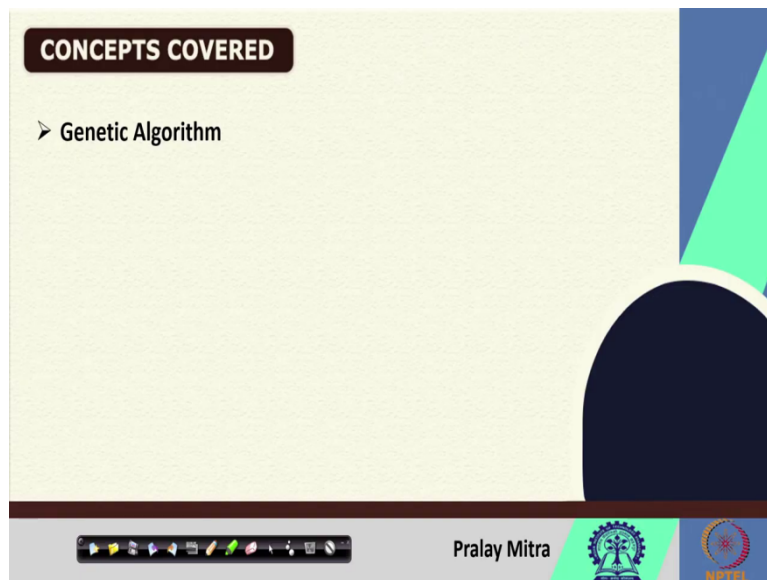


Algorithms for Protein Modelling and Engineering
Professor. Pralay Mitra
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur
Lecture No. 15
Genetic Algorithm (GA) for Surface Comparison

Welcome back, in this lecture, we will be focusing on another technique, which is completely different from the brute force technique or the geometric hashing-based technique that we discussed in the past weeks. This week we focus on the genetic algorithm for surface comparison.

(Refer to Slide Time: 0:40)

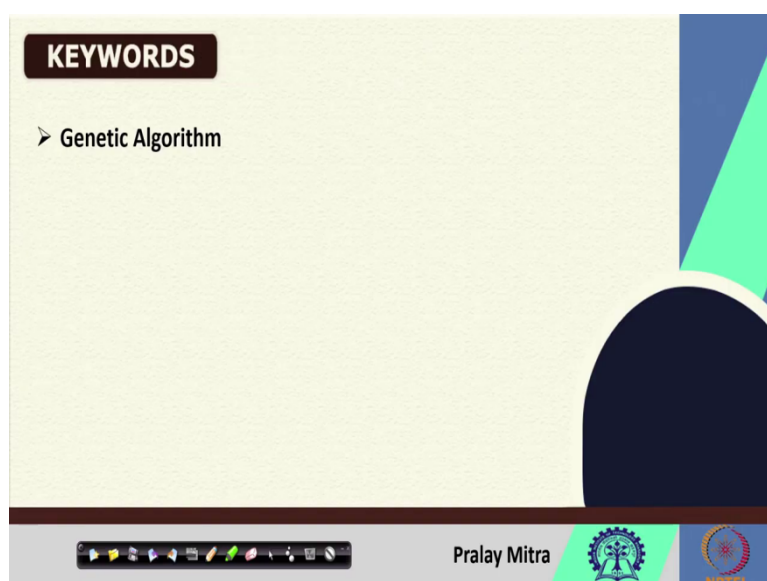


CONCEPTS COVERED

- Genetic Algorithm

Pralay Mitra

The slide features a light green background with a dark blue and light green geometric design on the right side. A dark blue semi-circle is at the bottom right. The footer contains a navigation toolbar, the name 'Pralay Mitra', and logos for IIT Kharagpur and NPTEL.



KEYWORDS

- Genetic Algorithm

Pralay Mitra

The slide features a light green background with a dark blue and light green geometric design on the right side. A dark blue semi-circle is at the bottom right. The footer contains a navigation toolbar, the name 'Pralay Mitra', and logos for IIT Kharagpur and NPTEL.

The concept you will cover is the genetic algorithm. We will not go into details, we will discuss this genetic algorithm only in the context of surface matching.

(Refer to Slide Time: 0:57)

Genetic Algorithm

For the computation intensive optimization problems, where deterministic algorithms may not provide the satisfactory solutions – non-deterministic algorithms like genetic algorithm (GA) attempts to mimic the processes of Darwinian evolution to finding the good, but not necessarily optimal solutions.

Pralay Mitra

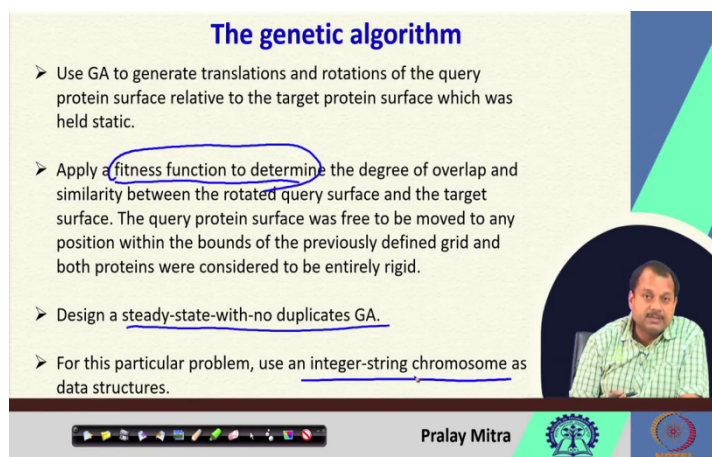
IIT Bombay NPTEL

This genetic algorithm concept was developed by borrowing the concept of Darwinian evolution. For computation-intensive optimization problems, where the deterministic algorithms may not provide satisfactory solutions, non-deterministic algorithms like a genetic algorithm which is also in short call as the GA, attempt to mimic the process of Darwinian evolution to find the good, but not necessarily optimal solution. I would like to highlight that it is a non-deterministic algorithm. Next, it mimics the process of Darwinism evolution for finding the good, but not necessarily optimal solution, which means that optimality is not guaranteed, but sometimes it can provide you a good solution. If you are utilizing a genetic algorithm, then, sometimes you will get the good but, not necessarily the optimal solution. That is why it is better to pick a list of probable solutions instead of one and from there using another level of filtering or another set of verification, you can identify the good solutions or better solutions compared to this genetic algorithm technique.

(Refer to Slide Time: 2:45)

The genetic algorithm

- Use GA to generate translations and rotations of the query protein surface relative to the target protein surface which was held static.
- Apply a fitness function to determine the degree of overlap and similarity between the rotated query surface and the target surface. The query protein surface was free to be moved to any position within the bounds of the previously defined grid and both proteins were considered to be entirely rigid.
- Design a steady-state-with-no duplicates GA.
- For this particular problem, use an integer-string chromosome as data structures.



Pralay Mitra

The genetic algorithm (GA) generates the translations and rotations of the query protein surface relative to the target protein surface which was held static. The same concept that we used for the brute force algorithm and geometric hashing technique. In the case of brute force, we decided one has to be static no change there. Another is the mobile. Usually for the computational advantage, we consider the smaller among these two is considered as a mobile molecule, but if both of them are of the same size, then arbitrarily you can choose one to be static and the other one will be mobile. The mobile molecule will rotate, and translate to generate several orientations.

For geometric hashing, we pick one for the offline calculation, that will generate the model and the basis points and that information will be stored in a two-dimensional hash bin. During the recognition phase, the voting will take place. Also for GA or genetic algorithm, translate and rotate the query protein surface related to the target protein surface, which was held static.

As of now, we are mentioning the protein surface. Now what kind of surface representation we did not mention? Shortly we will see this surface representation is the dense surface representation for better fitment and that is the Connolly surface that is used using Connolly surface calculation (lecture 14). We introduced the molecular surface and discussed in detail what is the difference between the accessible surface area and Connolly surface area which is computed by the summation of the contact area plus the re-entrant surface.

After generating the translations, and the different orientations using the translation and rotation, you need to apply a fitness function. That is the important part of any GA or genetic algorithm - the development of a fitness function to determine the degree of overlap and similarity between the rotated query surface and the target surface. This way it is similar to

the previous one. What are the similarities and dissimilarities between three different techniques - GA, geometric hashing, and the brute force technique?

After the generation phase, you need to score them. Also, you remember it can be integrated scoring or edge scoring. During the generation, you can score and consider whether this orientation is going to stay or go. I mean accepted or rejected or it can be considered or it can be eliminated or you can generate all the possibilities and then score them to check how many of them will retain.

Now, between the query and the target surface, the query protein surface was free to be moved from one position to any position within the bounds of the previously defined rate and both proteins are considered to be entirely rigid. The advantage is purely for the computation when the smaller molecule is considered as the mobile molecule.

Next, you design a steady state with no duplicate GA, so there are a lot of variations of the GA. Usually, this steady state with no duplicate GA is used for this kind of generation phase for this particular problem. Use an integer string chromosome as a data structure.

For this fitness function, you are matching the surface. First of all, it is the geometric feature, which you are trying to match. For enzyme-substrate binding, antigen-antibody, or other protein molecule binding, it is said that, whenever two protein molecules are interacting or are matching, then the surface should match. And there should be a kind of lock and key mechanism. Again lock and key mechanism is more successful in the enzyme-inhibitor or enzyme-substrate, but for the antigen-antibody or other protein molecules or protein and small molecule also you cannot rule out the geometric fitting or matching. For the fitting, the geometric feature will come into play along with the physicochemical information. In the Journal of molecular biology, Gabb et. al published one such work where the biochemical information is combined with the geometric information.

Then you will get a better matching or fitting and you can prune out or you can remove more false positive cases compared to if you considered either of them. All these are physicochemical or only geometric and I believe that you can understand also it is true. For this particular problem, we are using an integer string chromosome as a data structure. The good news is that, since it is an integer, you can go for an integer calculation rather than a floating point calculation which is a bit costly on the computer. Although at the programmer's level, you may not feel that.

(Refer to Slide Time: 9:26)

The genetic algorithm

- Each chromosome in the population contained six elements corresponding to the six degrees of freedom (3 rotation, 3 translation) necessary to completely move one rigid body relative to another.
- Randomly generate an initial population of chromosomes to transform query protein surface, which was first translated within the grid surrounding the target and then rotated about axes centered on its center of gravity.
- Evaluate fitness function of the GA to test the efficacy of the individual chromosomes. The chromosomes that produced better solutions being given a higher fitness and vice versa.

PRN
↓
pseudorandom
number

Pralay Mitra

Since the concept is borrowed from Darwin's evolution, the keyword chromosome is being used, but you can consider it as a fingerprint that contains the feature information. The data structure is an integer in a population that contains six elements corresponding to the six degrees of freedom (3 rotation+3 translation). It is necessary to completely move one rigid body related to another.

Randomly generate an initial population of chromosomes to transform query protein surface, which was first translated within the grid surrounding the target and then rotated about the axis centered on its center of gravity. To generate the initial population, the chromosome values have been created randomly. Hence, a chromosome is an integer data structure and in that data structure, 3 translation and 3 rotation information are incorporated. We know by this time that if we have 3 translation and 3 rotation information which means, along the x-axis along the y-axis along the z-axis about the x-axis about the y-axis about the z-axis, then I can get the transformation matrix for transforming one protein molecule from one position to another position.

To generate the initial population, randomly populate that chromosome, which will give some random transformation. A random number of translations and rotation or random transformation is given, which was translated within the grid surrounding. Now for each such situation, you evaluate the fitness function of the GA to test the efficacy of the individual chromosome. Initially, you are starting with the random but as the simulation progresses, then it will not be random. However, this evaluation of the fitness function will always take place. In the beginning, when it's random, you will see a lot of situations will be ruled out, but still, few are supposed to be there. That's why during the random number generation you should

be careful about using a proper random number or if not then at least some pseudo-random number (PRN).

We will see in detail when we will go to the Monte Carlo simulation or a replica exchange Monte Carlo simulation that there is a role for this random number generation. If it is not sufficiently random, then there may be some biases and because of that bias, a result may vary a lot. That's why extra care should be taken when you are randomly generating an initial population.

Sometimes the inbuilt functions may not be sufficient for generating the random number. I am talking about the implementation - whatever may be the language. You have to tweak it sometimes to make it more random so that the pattern occurs at a large interval and you are within a safe region.

Evaluate the fitness function of the GA to test the efficacy of the individual chromosomes. The chromosome that produced the better solution is given a higher fitness and vice versa. You are selecting based on that fitness function that is more probable, as we mentioned in the definition part of this GA. It may not give you the optimal solution, but a good solution, and we are looking for that good solution using this technique.

(Refer to Slide Time: 13:40)

The slide is titled "The genetic algorithm" in blue text. It contains three bullet points:

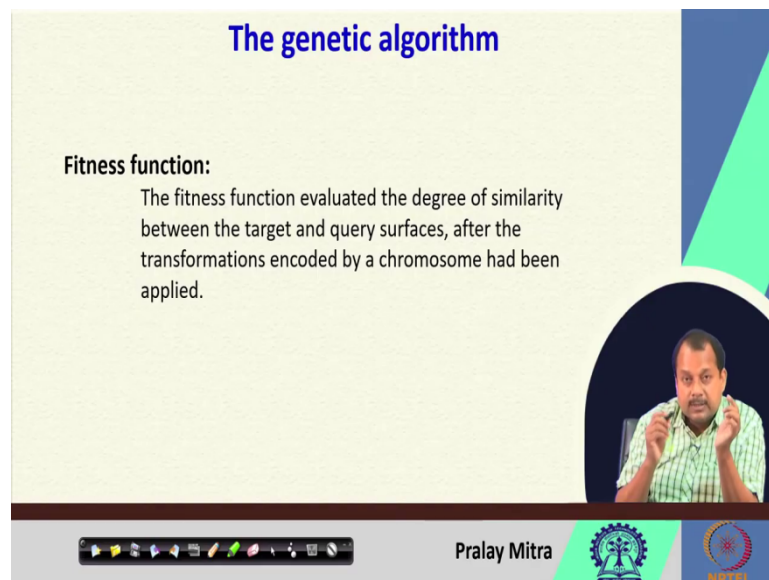
- Run a series of cycles in GA where at each iteration a subset of members of the existing population is replaced with an equal number of new members
- Evaluate the fitness of the new members aiming to increase the average fitness of the population and hence move towards the optimal solution to the problem.
- The replacement of population members was controlled by the use of three genetic operators, i.e., parental selection, crossover, and mutation.

At the bottom right of the slide, there is a video inset showing a man, Pralay Mitra, speaking. Below the slide, there is a toolbar with various icons and the name "Pralay Mitra" next to the NPTEL logo.

Next, run a series of cycles in GA, where at each iteration a subset of numbers of the existing population is replaced with an equal number of new numbers. When these new numbers are being introduced, you make sure that there is a variation in that one so that you can pick different unbiased possibilities among them.

Subsequently, evaluate the fitness of the new members aiming to increase the average fitness of the population and hence move towards the optimal solution to the problem. We are trying to reach the optimal solution, but it may not be guaranteed always. The replacement of population members was controlled by the use of three genetic operators, that is parental selection, crossover, and mutation. These three concepts again are taken from evolutionary biology. The parental selection process, then crossing over of the chromosomes, so that some part of the previous one is coming to the next and mutations. Hence, not all the three members, I mean 3 translations then 3 rotations you are going to change but, selectively you are changing and during the selection process all 6 may be changed, but starting with one. Randomly you choose and change, randomly you choose and change that way you go for some mutation.

(Refer to Slide Time: 15:19)



The genetic algorithm

Fitness function:
The fitness function evaluated the degree of similarity between the target and query surfaces, after the transformations encoded by a chromosome had been applied.

Pralay Mitra

NPTEL

It completes the entire process. To summarize that we define one fitness function and that fitness function evaluated the degree of similarity between the target and query surface after the transformations encoded by chromosome have been applied. Now, this fitness function is at the core, based upon the goodness of your design of this fitness function, not only that but yes, that is one of the primary criteria, the goodness of your simulation will depend and also the convergence of your simulation will depend, so what you are doing? You are defining one fitness function then you are defining one chromosome that is your data structure - in our case, it is an integer chromosome. You can have only the 6 degrees of freedom information, I mean 3 translations, and 3 rotations or you can have other information also with that one. Now, using that chromosome information, you initially populate the chromosome and during

that population, you randomly generate some chromosomes. Then using the fitness function, you pick what will lead to an optimal solution.

After that at each step, you reject some previous ones and incorporate some new ones and during this process, parental selection crossover and the mutation of those three will be a crucial component of your design. So, combining this you are having one simulation environment which is non-deterministic and that is called the genetic algorithm.

(Refer to Slide Time: 17:06)

The genetic algorithm

Choice of GA parameter settings:

- Population size
- Number of runs
- Percentage of population replaced each iteration (%) $10 \text{ pts}/\text{Å}^2$
- Mutation/crossover rate (%) ✓
- Selection pressure
- Niche sphere diameter (Å)
- Angle allowed between normals (°)
- Number of niches created
- Creep range for rotation (°) 1
- Creep range for translation (Å) 0.5 Å

Pralay Mitra

Once that algorithm is designed, then you have to tune that algorithm based on some known data so that your feature set is good enough for this particular purpose. When you are selecting the parameters for the setting of the GA algorithm, what you have to decide is the population size. How many populations will be there? The number of runs, how many times you will keep on running? Percentage of the population replaced in each iteration.

You can go up to the replacement of all the population or a percentage (>0) of the population. Now, the mutation or crossover rate that I mentioned is in total 6. Starting from one mutation, you can mutate 1/2/3/4/5/all 6. You can mutate only the translation part or rotation part or better you can decide that randomly. I will rotate a random number of positions, which means 6 positions are there, so I will generate two random numbers, one will tell me how many mutations I will perform so that the first random number will vary from one to 6. Then the second random number will be telling me at which point(s) I will change. If the first random

number is 1, then the second random number will tell out of 6 positions which one I will mutate.

If the first random number is 3, then the second, three more random numbers I will generate. They will be different and will vary from 1 to 6. Similarly, you have to decide about the mutation or crossover rate, it can be fixed, or it can be dynamic during the simulation.

Then you can decide on selection pressure. Niche sphere dimension in the angstrom angle allowed between normal's and when it is normal, then you remember that in Connolly's angle's algorithm, we consider 10 points per angstrom square. There 10 points are surface normals.

How many angles I can allow to deviate between the normal that I can decide. Several niche created, creep range of rotation, and creep range of translation, so usually for a good result you can think of 0.5Å for the translation, and for this rotation, you can consider 1Å sorry not Å 1°, 1° degree of rotation with 360 steps.

If it is 0.5 Å based upon the size of the protein or the number of grids with this that will vary and in the brute force algorithm, if you remember what we did. I suggested that it will be something between 6° to 12° and even up to 18° not more than 18° or not less than 6°. Translation also, we probably decided 1.8Å that way we can reduce. If I go with this one then it will be a very high number of steps or a large number of steps, then you have to perhaps think of parallel implementation. Otherwise, it will be time-consuming. That's it about the genetic algorithm.

(Refer to Slide Time: 21:37)

The slide is titled "A comparison study" and lists four techniques:

- > Brute force technique
- > FFT based improvement
- > Geometric Hashing
- > Genetic Algorithm

Handwritten notes in blue ink are present on the slide:

- Next to "Brute force technique": α, β, γ
- Next to "FFT based improvement": 360°
- Next to "Geometric Hashing": 1.8 \AA and 20.5 \AA
- Next to "Genetic Algorithm": $3^\circ, 5^\circ, 9^\circ, 12^\circ, 18^\circ$ and 10°

A small video inset in the bottom right corner shows a man speaking. The slide footer includes the name "Pralay Mitra" and logos of institutions.

If I go for the comparison study, let us start with the last genetic algorithm that we discussed just now. Long back when the genetic algorithm concept was introduced, and people are

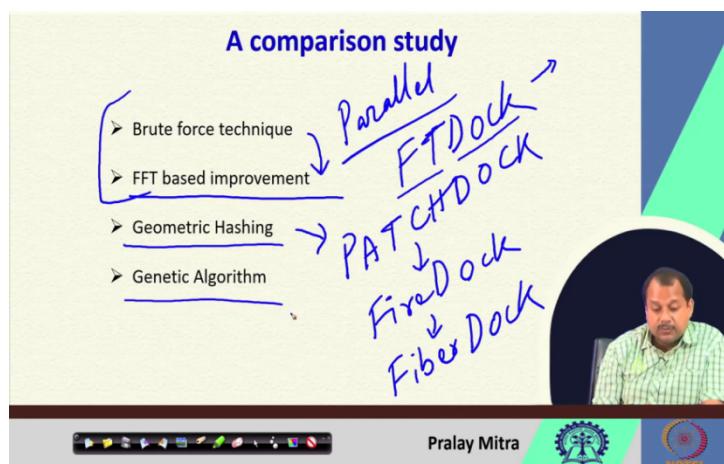
trying to implement it, but didn't get that much popularity in the case of surface matching or as an extension problem that surface matching to protein docking decoy generation. It did not get that much success on that one. On the other hand brute force technique, as the name suggests is very useful in the sense that, it can generate almost all the possibilities and that you can control by deciding what will be the grid step and angle of rotation.

Usually, this grid state is 1.8Å and the angle of rotation can be 6° or 9° or 12° or 18°. You can go up to 3°. But it is not suggested to go beyond 18°. You can consider 10° also. But one limitation is, you should decide the angle in such a way that if you divide 360° by that number, then you will get one integer value. That is the only criterion. Now for all these degrees 3, 6, 9, 12, 18, and 10, it will give an integer value, and since it's an integer value, so no problem. But, do not go below 3° or above 18°. Above 18° it will be a very coarse level orientation calculation, which will rule out a lot of positive cases. On the other hand, if it will be <3° then the number of orientations will increase heavily. However, if you plan for implementing in any parallel environment, then perhaps you can consider it.

Regarding the grid step, 1.8Å is good enough. You can reduce it to 0.5Å. Again, if you wish to increase from 1.8Å then you should be careful about the fact that van der Waals radius sulfur is 1.8Å to 1.85Å. Now, if you go beyond 1.8Å, then there is a possibility that in one grid cell, more than one atom will be accommodated. If that is the situation then along with the surface atom, another non-surface atom will also be accommodated. If the second atom is not on the surface then it is supposed to give you a penalty indicating that there is a penetration now will be absent. My suggestion is not to go to some large value for the grid step, but to keep these 1.8Å or some lesser value. Again not very less than 0.5Å, which will not give you much advantage.

When you are looking for the overlapping, if they are close enough to each other then also, they may be in a separate cell. Then our multiplication by the assumption that 1 indicates on the surface and 0 indicates on the outside and penalty is some high negative value or small positive value will not give you the correct result.

(Refer to Slide Time: 26:33)



Once you decide on that then you can use this FFT-based improvement. This FFT-based improvement is one of the masterwork in this context, where you can go for the brute force technique. And for the brute force technique, you have complete control over the computation time because you are going for the FFT-based improvement. Also, there is a possibility that you can go for parallel implementation for this, and also for all others. You can see that there is a possibility of parallelization and if you exploit that one then some time improvement will also be there. If one algorithm guarantees or suggests that my accuracy will be very high, but my computation time will also be high then my suggestion is to go for it. Because it guarantees some good results. Currently, the computation time is not a problem, even if you consider the storage for the geometric hashing then also it is not much. So, go for it.

The fast Fourier transform-based docking (FTDock) technique was the first algorithm, which was designed based on this approach. From this FTdock, some other docking techniques are also derived and they uses the same kind of concept as the FTdock utilizes. That way this is one of the pioneering works. After that one this geometric hashing-based technique is also developed and nurtured by the Wolfson group and then there are algorithms like PATCHDOCK. PATCHDOCK is the algorithm based on this geometric hashing technique. Now, this PATCHDOCK is upgraded to FireDock and it is further upgraded to FiberDock when they incorporated the flexibility in the protein backbone structure. It is a very good work from the group of Wolfson. Considering that the success rate for the genetic algorithm is not much.

In summary, the comparison of the techniques that we have discussed as of now, the references I have provided, wherever I have used some of the literature mostly, and since it is one of the advanced topics, most of the textbook does not have this kind of algorithm.

(Refer to Slide Time: 29:45)

REFERENCES

Poirrette, Andrew R., et al. "Comparison of protein surfaces using a genetic algorithm." *Journal of Computer-Aided Molecular Design* 11.6 (1997): 557-569.

The slide features a dark blue header with the word 'REFERENCES' in white. The main content area is light green. A circular video inset in the bottom right shows a man in a green and white checkered shirt. At the bottom, there is a navigation bar with various icons and two logos: a gear with a tree and the acronym 'NPTEL'.

And here is the reference for the genetic algorithm. It is a comparison of protein surfaces using a genetic algorithm published long back, in 1997. That's it. Thank you very much.