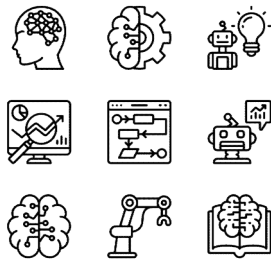


Computer Science for Practicing Engineers

Thuật toán heuristic

Thuật toán xác suất



TS. Huỳnh Bá Diệu

Email: dieuhb@gmail.com

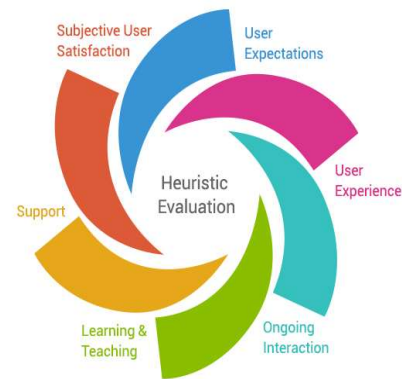
Phone: 0914146868

Nội dung

1. Thuật toán heuristic là gì?
2. Thuật toán xác suất là gì?
3. Phân lớp thuật toán xác suất

Thuật toán heuristic

A heuristic algorithm is one that is designed to solve a problem in a faster and more efficient fashion than traditional methods by sacrificing optimality, accuracy, precision, or completeness for speed.



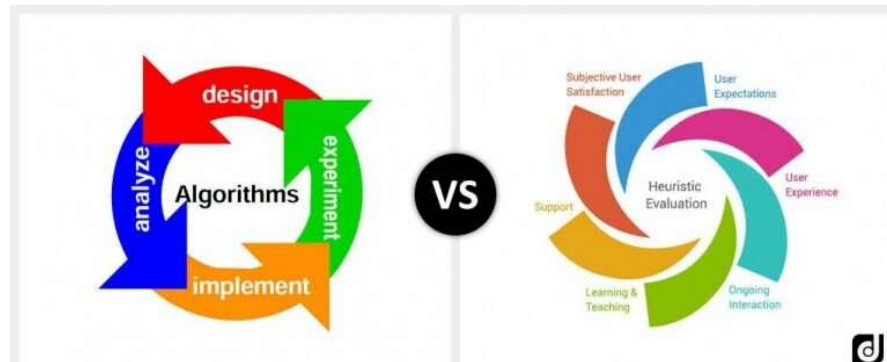
Thuật toán heuristic

Heuristic algorithms are often used to solve NP-complete problems, a class of decision problems. In these problems, there is no known efficient way to find a solution quickly and accurately although solutions can be verified when given.

Heuristics can produce a solution individually or be used to provide a good baseline and are supplemented with optimization algorithms.

Thuật toán heuristic

Heuristic algorithms are most often employed when approximate solutions are sufficient and exact solutions are necessarily computationally expensive.

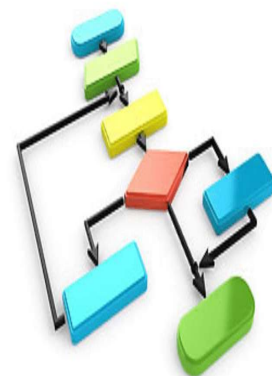


Thuật toán heuristic

Thuật toán: là dãy xác định các câu lệnh đảm bảo việc tìm ra lời giải cho bài toán.

Thuật toán heuristic: tập các qui luật, kinh nghiệm khi giải quyết bài toán. Thuật toán heuristic không đảm bảo việc cho ra lời giải đúng.

Thuật toán HA **được suy ra từ kinh nghiệm**, cho phép tìm ra **lời giải có thể chấp nhận được** nhưng không chứng minh tính đúng đắn hay tối ưu.



ComputerHope.com

Thuật toán heuristic

Thuật toán heuristic bỏ qua tiêu chí tối ưu hay biết rõ thời gian, tài nguyên khi thực hiện thuật toán.

Thuật toán AH chỉ tìm ra giải pháp chấp nhận được, không chứng minh được nó tối ưu hơn giải pháp khác.

Ví dụ: Bài toán mua hay thuê nhà.



Thuật toán heuristic

Có những yếu tố ảnh hưởng đến bài toán mà ta không thể định giá được khi xây dựng thuật toán.

Thuật toán H dù không cho ra lời giải tối ưu nhưng vẫn sử dụng vì nó làm đơn giản quyết định và cho ra lời giải đủ tốt.

Theo các bạn, các yếu tố nào trong bài toán mua nhà không biểu diễn được trong máy tính???



Thuật toán heuristic

Ví dụ bài toán người du lịch qua n thành phố.

Dùng vét cạn: không thể thử hết $(n-1)!$ các trường hợp.

Nếu $n = 21$ thì thử $(20)!$ trường hợp

$= 2,432,902,008,176,640,000$

Dùng H thì độ phức tạp là $O(n^2)$

Vì sao là $O(n^2)$???

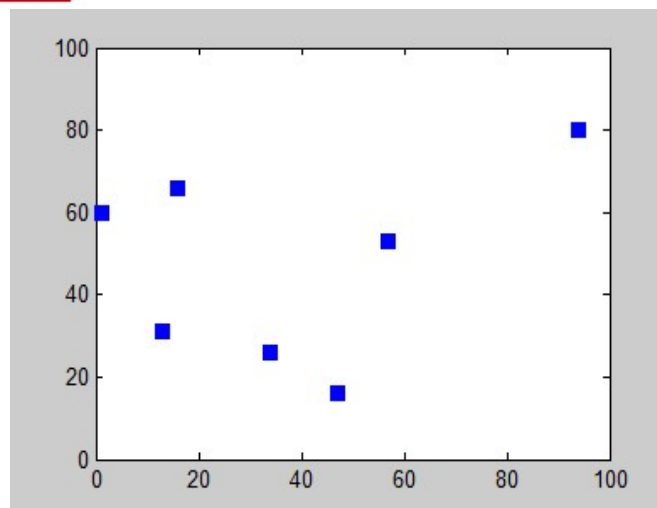
Vì sao vét cạn là $O(n!)$???



Thuật toán heuristic: Traveling Salesmen Problem

Nearest neighbor (NN)

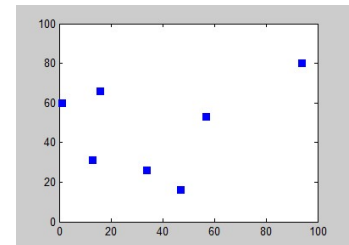
Starting from a randomly chosen city, the algorithm finds the closest city. The remaining cities are analyzed again, and the closest city is found.



Thuật toán heuristic: **Traveling Salesmen Problem**

Thuật toán Nearest neighbor

1. Start at a random vertex
2. Determine the shortest distance connecting the current vertex and an unvisited vertex V
3. Make the current vertex the unvisited vertex V
4. Make V visited, record the distance traveled
5. Terminate if no other unvisited vertices remain
6. Repeat step 2



Thuật toán heuristic: **Traveling Salesmen Problem**

There are 4 points of interest located in a 10x10 plot of space: (3,4.5), (9,6.25), (1,8), and (5.5,0). The table below lists the distance required to touch all 4 points with the first and last point known using the nearest neighbor algorithm:

		Ending point			
		(5.5,0)	(1,8)	(3,4.5)	(9,6.25)
Starting Point	(5.5,0)	0	9.18	5.15	7.16
	(1,8)	9.18	0	4.03	8.19
	(3,4.5)	5.15	4.03	0	6.25
	(9,6.25)	7.16	8.19	6.25	0

Thuật toán heuristic: **Traveling Salesmen Problem**

Starting at point (1,8): The shortest distance to an unvisited point is 4.03 units to point (3,4.5). The shortest distance to an unvisited point is 5.15 units to point (5.5,0). The shortest distance to an unvisited point is 7.16 units to point (9,6.25). The total distance traveled is 16.34 units.

		Ending point			
		(5.5,0)	(1,8)	(3,4.5)	(9,6.25)
Starting Point	(5.5,0)	0	9.18	5.15	7.16
	(1,8)	9.18	0	4.03	8.19
	(3,4.5)	5.15	4.03	0	6.25
	(9,6.25)	7.16	8.19	6.25	0

Thuật toán heuristic: **Traveling Salesmen Problem**

Starting at point (9,6.25): The shortest distance to an unvisited point is 6.25 units to point (3,4.5). The shortest distance to an unvisited point is 4.03 units to point (1,8). The shortest distance to an unvisited point is 9.18 units to point (5.5,0). The total distance traveled is 19.46 units. but not the best solution.

		Ending point			
		(5.5,0)	(1,8)	(3,4.5)	(9,6.25)
Starting Point	(5.5,0)	0	9.18	5.15	7.16
	(1,8)	9.18	0	4.03	8.19
	(3,4.5)	5.15	4.03	0	6.25
	(9,6.25)	7.16	8.19	6.25	0

Thuật toán heuristic: **Traveling Salesmen Problem**

Both situations followed the NN algorithm to solve the problem, however the total distance traveled changed based on the started location. This shows how a heuristic algorithm can give a good solution.

		Ending point			
		(5.5,0)	(1,8)	(3,4.5)	(9,6.25)
Starting Point	(5.5,0)	0	9.18	5.15	7.16
	(1,8)	9.18	0	4.03	8.19
	(3,4.5)	5.15	4.03	0	6.25
	(9,6.25)	7.16	8.19	6.25	0

Thuật toán xác suất (probabilistic algorithms)

Thuật toán đơn định: với 1 dữ liệu vào chỉ có duy nhất một dữ liệu ra

Thuật toán xác suất là thuật toán mà kết quả của nó phụ thuộc vào một số ngẫu nhiên.

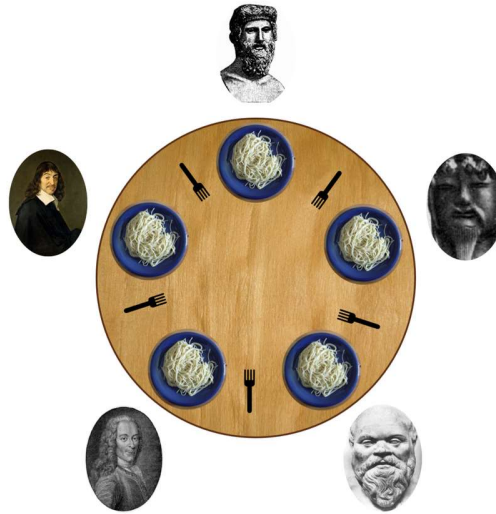
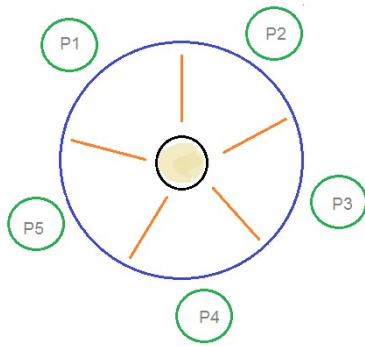
Thuật toán xác suất còn gọi là thuật toán ngẫu nhiên.

Hoạt động của thuật toán có thể khác nhau trên cùng một dữ liệu vào.

Vì sao hoạt động khác nhau???

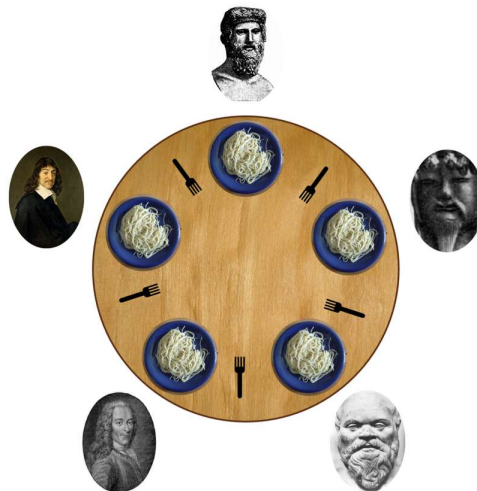
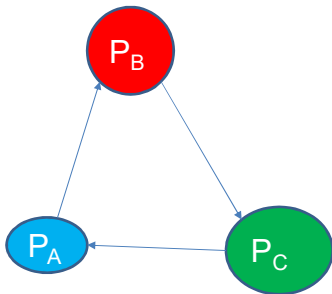


Thuật toán xác suất: bữa ăn của 5 nhà triết học



Thuật toán xác suất: bữa ăn của 5 nhà triết học

Hệ điều hành: Quản lý tài nguyên găng



Thuật toán xác suất: probabilistic algorithms

- Ưu điểm:

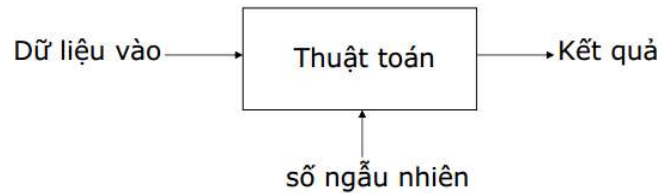
- Đơn giản

- Hiệu quả

- Nhược điểm:

- Có thể cho kết quả sai

- Hoặc không dừng



Thuật toán xác suất: probabilistic algorithms

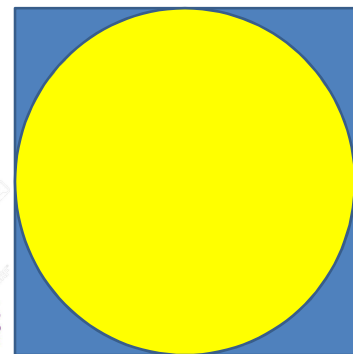
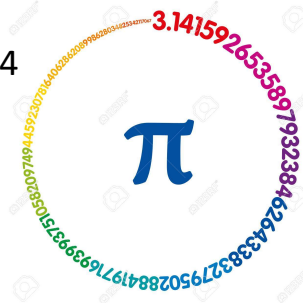
Ví dụ: Tính gần đúng số pi

Cho hình vuông có độ dài cạnh a

Hình tròn nội tiếp hình vuông có bán kính $a/2$

Diện tích hình vuông là a^2

Diện tích hình tròn là $(a^2 * \pi) / 4$



Thuật toán xác suất: probabilistic algorithms

Kiểm tra điểm thuộc đường tròn:

$$x^2 + y^2 \leq 1$$

```
x= ((double)rand() / (double)(RAND_MAX));
```

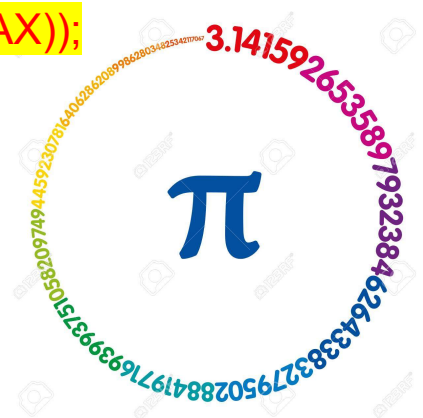
```
y= ((double)rand() / (double)(RAND_MAX));
```

Gọi hàm main:

N=1 000

N=1 000 000

N=1 000 000 000



Thuật toán xác suất: probabilistic algorithms

```
int main() {
    long n;
    n=1000; cout<<"\n n="<<n<<" PI= "<< TinhPi(n);
    n=1000000; cout<<"\n n="<<n<<" PI= "<< TinhPi(n);
    n=1000000000; cout<<"\n n="<<n<<" PI= "<< TinhPi(n);
}
```

Thuật toán xác suất: probabilistic algorithms

```
double tinhPi(int n) {
    double k=0;
    for (int i=1; i<=n; i++)
    {
        x= ((double)rand() / (double)(RAND_MAX));
        y= ((double)rand() / (double)(RAND_MAX));
        if(x*x + y*y <=1) k++;
    }
    return 4*k/n;
}
```

Thuật toán xác suất: probabilistic algorithms

Cho dãy A gồm n phần tử. Dãy A được gọi là dãy chứa phần tử đa số nếu như một phần tử nào đó trong mảng chiếm hơn nửa.

Hãy kiểm tra mảng a có phải đồng đức không.

Dữ liệu vào FILE DD.inp chứa các số nguyên

Dữ liệu ra FILE DD.out chứa số 0 hoặc 1.

Thuật toán xác suất: probabilistic algorithms

Cho dãy A gồm n phần tử. Dãy A được gọi là dãy chứa phần tử đa số nếu như một phần tử nào đó trong mảng chiếm hơn nửa.

- Cách tiếp cận bình thường: SOL1()
- Cách tiếp cận theo thuật toán xác suất: SOL2()

Thuật toán xác suất: probabilistic algorithms

Cho dãy A gồm n phần tử. Dãy A được gọi là dãy chứa phần tử đa số nếu như một phần tử nào đó trong mảng chiếm hơn nửa.

- Cách tiếp cận bình thường: SOL1()

1. Sắp xếp
2. Cho i từ 1 đến $n/2 - 1$
 nếu $a[i] = a[i+n/2]$ return 1
3. return 0

[4 2 5 4 7 5 5 8 3 4 5 5 1 5 5 5 2 5 5 5 5 1 5 5 8 4 6 5 5 5 5 5]

Phân lớp các Thuật toán xác suất (Classification of PA)

There is a variety of behaviours associated with probabilistic algorithms:

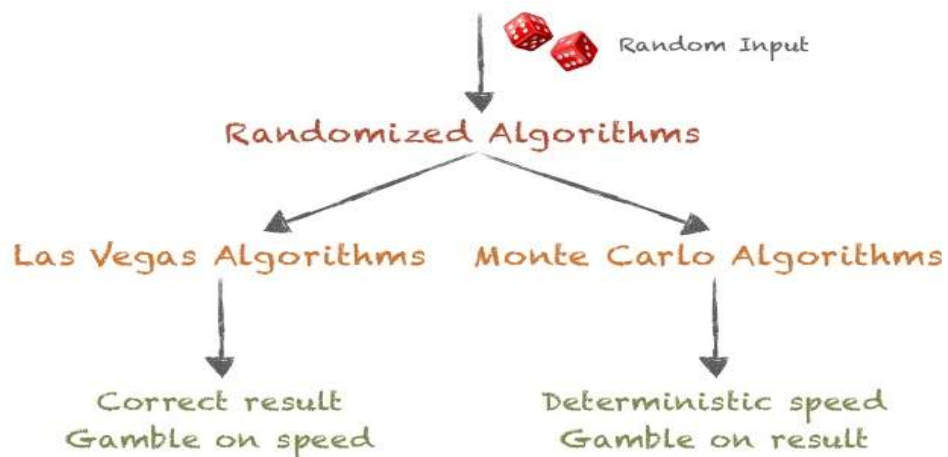
1/ Algorithms which *always return a result, but the result may not always be correct*. We attempt to minimize the probability of an incorrect result, and using the random element, multiple runs of the algorithm will reduce the probability of incorrect results. These are sometimes called Monte Carlo algorithms.

Phân lớp các Thuật toán xác suất (Classification of PA)

There is a variety of behaviours associated with probabilistic algorithms:

2/ Algorithms that *never return an incorrect result, but may not produce results at all on some runs*. Again, we wish to minimize the probability of no result, and, because of the random element, multiple runs will reduce the probability of no result. These are sometimes called Las Vegas algorithms.

Phân lớp các Thuật toán xác suất (Classification of PA)



Thuật toán xác suất: probabilistic algorithms

Example: finding an ' a ' in an [array](#) of n elements.

- **Input:** An array of $n \geq 2$ elements, in which half are ' a 's and the other half are ' b 's.
- **Output:** Find an ' a ' in the array.

We give two versions of the algorithm, one [Las Vegas algorithm](#) and one [Monte Carlo algorithm](#).

Thuật toán xác suất: Las Vegas algorithm

findingA_LV(array A, n)

begin

repeat

Randomly select one element out of n elements.

until 'a' is found

end

This algorithm succeeds with probability 1. The number of iterations varies and can be arbitrarily large, but the expected number of iterations is:

$$\lim_{n \rightarrow \infty} \sum_{i=1}^n \frac{i}{2^i} = 2$$

Since it is constant the expected run time over many calls is Theta(1).

https://en.wikipedia.org/wiki/Randomized_algorithm

Thuật toán xác suất: Monte Carlo

findingA_MC(array A, n, k)

begin

i=0

repeat

Randomly select one element out of n elements; i = i + 1;

until i=k or 'a' is found

end

If an 'a' is found, the algorithm succeeds, else the algorithm fails.

After k iterations, the probability of finding an 'a' is:

$$\Pr[\text{find a}] = 1 - (1/2)^k$$

Thuật toán xác suất: Monte Carlo

findingA_MC(array A, n, k)

begin

end

If an ' a ' is found, the algorithm succeeds, else the algorithm fails.
After k iterations, the probability of finding an ' a ' is:

$$\Pr[\text{find } a] = 1 - (1/2)^k$$

This algorithm does not guarantee success, but the run time is bounded. The number of iterations is always less than or equal to k . Taking k to be constant the run time (expected and absolute) is $\Theta(1)$.

Thuật toán xác suất: Las Vegas algorithm

Randomized QuickSort(S)

When the input S is stored in an array A

```
QuickSort( $A, l, r$ )
{
  If ( $l < r$ )
     $x \leftarrow$  an element selected randomly uniformly from  $A[l..r]$ ;
     $i \leftarrow \text{Partition}(A, l, r, x)$ ;
    QuickSort( $A, l, i - 1$ );
    QuickSort( $A, i + 1, r$ )
}
```

Hãy cài thuật toán Quicksort với các giá trị chốt khác nhau, so sánh thời gian các lần chạy??? ($n=1000000$)

Thuật toán xác suất: Monte Carlo

Approximate median

Definition: Given an array $A[]$ storing n numbers and $\epsilon > 0$, compute an element whose rank is in the range $[(1 - \epsilon)n/2, (1 + \epsilon)n/2]$.

Thuật toán xác suất: Monte Carlo

Approximate median

A Randomized Algorithm:

1. Select a random sample S of $O(\frac{1}{\epsilon} \log n)$ elements from A .
2. Sort S .
3. Report the median of S .

Running time: $O(\frac{1}{\epsilon} \log n \log \log n)$

The output is an ϵ -approximate median with probability n^{-2} .

For $n \sim$ a million, the error probability is 10^{-12} .

Thuật toán xác suất

Thảo luận nhóm về các thuật toán kiểm tra số nguyên n có phải số nguyên tố không!!!

Trình bày về thuật toán **Miller-Rabin** (theo tiếp cận thuật toán xác suất)

Tham khảo tại thêm tại:

<http://www.giaithuatlaptrinh.com/?p=278>



Tài liệu đọc thêm

<https://www.geeksforgeeks.org/dining-philosopher-problem-using-semaphores/>

<https://www.geeksforgeeks.org/randomized-algorithms/>

Link YouTube

<https://www.youtube.com/watch?v=9f1oOMX3mP4>

<https://www.khanacademy.org/computing/computer-science/cryptography/random-algorithms-probability/v/randomized-algorithms-prime-adventure-part-8>

